Data Stream and Object Architectures

Bar Code Object Content Architecture Reference

S544-3766-05

Data Stream and Object Architectures

Bar Code Object Content Architecture Reference

S544-3766-05

Note!
Before using this information and the product it supports, read the information in "Notices" on page 117.

Sixth Edition (November 2003)

This edition applies to IBM Bar Code Object Content Architecture until otherwise indicated in new editions or technical newsletters. This edition replaces S544-3766-04.

| Changes are indicated by a vertical bar to the left of the change. For a detailed list of changes, refer to "Changes in This Edition" on page ix.

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

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

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

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

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

Preface

This book describes the functions and services associated with Bar Code Object Content Architecture™ (BCOCA™).

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

Who Should Read This Book

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

How to Use This Book

This book is divided into six chapters and three appendixes:

- Chapter 1, "A Presentation Architecture Perspective," on page 1 introduces IBM's presentation architectures and describes the role of data streams and data objects.
- Chapter 2, "Introduction to BCOCA," on page 9 describes bar code symbols, bar code symbologies, and the basic elements of a bar code system.
- Chapter 3, "BCOCA Overview," on page 17 describes the key concepts of the BCOCA architecture and its relationship to other presentation architectures.
- Chapter 4, "BCOCA Data Structures," on page 25 defines the data structures, fields, and valid data values assigned to and reserved for the BCOCA architecture.
- Chapter 5, "Exception Conditions," on page 101 lists the exceptions to the BCOCA definitions and what to do when such exceptions occur.
- Chapter 6, "Compliance," on page 107 describes how products may be valid generators or receivers of a BCOCA object.
- Appendix A, "Bar Code Symbology Specification References," on page 109 lists the bar code symbology specifications referenced in this document.
- Appendix B, "MO:DCA Environment," on page 111 describes how BCOCA bar code objects are defined and used in the MO:DCA™ environment.
- Appendix C, "IPDS Environment," on page 113 describes how BCOCA bar code objects are defined and used in the IPDS™ environment.

The "Glossary" on page 121 defines terms used within the book.

How to Read the Syntax Diagrams

Throughout this book, syntax for the BCOCA data structures is described using the structure defined in Table 1.

Table 1. Data Structure Syntax

Offset	Type	Name	Range	Meaning	BCD1 Range
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	Subset of the range of values that must be supported by all BCOCA receivers. Refer to Chapter 6, "Compliance," on page 107 for additional details.

The four basic data types used in BCOCA syntax tables are:
CODE Architected constant
BITS Bit string
UBIN Unsigned binary
UNDF Undefined data type

The following is an example of a BCOCA data structure:

Offset	Type	Name	Range	Meaning	BCD1 Range
0	BITS	Flags			
bit 0		HRI	B'0' B'1'	HRI is presented HRI not presented	B'0' B'1'
bits 1–2		Position	B'00' B'01' B'10'	Default HRI below HRI above	B'00' B'01' B'10'
bit 3		SSCAST	B'0' B'1'	Asterisk is not presented Asterisk is presented	B'0' B'1'
bit 4			B'0'	Reserved	
bit 5		Suppress bar code symbol	B'0' B'1'	Bar code symbol suppression: Present symbol Suppress symbol	B'0'
bit 6		Suppress blanks	B'0' B'1'	Desired method of adjusting for trailing blanks: Don't suppress Suppress and adjust	B'0'
bit 7			B'0'	Reserved	
1–2	UBIN	Xoffset	X'0001'–X'7FFF'	X_{bc}-coordinate of the symbol origin in the bar code presentation space	X'0001'–X'7FFF' Refer to the note following the table.
3–4	UBIN	Yoffset	X'0001'–X'7FFF'	Y_{bc}-coordinate of the symbol origin in the bar code presentation space	X'0001'–X'7FFF' Refer to the note following the table.
The following special-function information is only used with the following bar code types: Data Matrix, MaxiCode, PDF417, QR Code					
5–n		Special functions	See field description	Special-function information that is specific to the bar code type	Not supported in BCD1
The following symbol data is specified for all bar code types					
n+1 to end	UNDF	Data	Any value defined for the bar code type selected by the BSD	Data to be encoded	Any value defined for the bar code type selected by the BSD

Note: The BCD1 range for these fields have been specified assuming a unit of measure of 1/1440 of an inch. Many receivers support the BCD1 subset plus additional function. If a receiver supports additional units of measure, the BCOCA architecture requires the receiver to at least support a range equivalent to the BCD1 range relative to each supported unit of measure. More information about supported-range requirements is provided in the section titled "L-unit Range Conversion Algorithm" on page 20.

Notation Conventions

The following notation conventions apply to the BCOCA data structures.

- Each byte contains eight bits.
- Bytes of a BCOCA data structure are numbered beginning with byte 0. For example, a two-byte field followed by a one-byte field would be numbered as follows:
  Bytes 0–1
  Field 1
  Byte 2 Field 2
- Bit strings are numbered beginning with 0. For example, a one-byte bit string contains bit 0, bit 1, ..., bit 7.
- Field values are expressed in hexadecimal or binary notation:

  X'7FFF' = +32767

  B'0001' = 1
- Some bits or bytes are labeled *reserved*. The content of reserved fields is not checked by BCOCA receivers. However, BCOCA generators should set reserved fields to the specified value, if one is given, or to zero.
- Values not explicitly defined in the range column of a field are reserved.
- Additional information about specific fields is listed after each data structure table.
- The term *default* is used in the description of some bits or bytes in the meaning column of the data structure tables. The default values for these fields are described in the field descriptions that follow the data structure tables.

Related Publications

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

IBM Architecture Publications

Title	Order Number
Bar Code Object Content Architecture Reference	S544-3766
Font Object Content Architecture Reference	S544-3285
Image Object Content Architecture Reference	SC31-6805
Intelligent Printer Data Stream™ Reference	S544-3417
Graphics Object Content Architecture Reference	SC31-6804
Graphics Object Content Architecture for Advanced Function Presentation™ Reference	S544-5498
Mixed Object Document Content Architecture™ Reference	SC31-6802
Presentation Text Object Content Architecture Reference	SC31-6803

You can order any of these architecture publications separately, or order them (except for S544-5498) as a group using SBOF-6179.

Title	Order Number
Character Data Representation Architecture Overview	GC09-2207
Character Data Representation Architecture Reference and Registry	SC09-2190

IBM ImagePlus Publications

Title	Order Number
IBM SAA® ImagePlus® Online Library CD-ROM	SK2T-2131
ImagePlus MVS/ESA™ General Information Manual	GC31-7537
AS/400® ImagePlus General Information Manual	GC38-2027
SAA ImagePlus/2 General Information Manual	GC28-8173

IBM Graphics and Image Publications

Title	Order Number
GDDM®, 5748-XXH: General Information Manual contains a comprehensive overview of graphics and image support for MVS™, VM, VSE, and OS/400® systems.	GC33-0100
Introducing GDQF contains a comprehensive overview of Graphic Query and Display Facilities for complex manufacturing graphics, image, and publishing products.	GH52-0249
OS/2® Presentation Manager GPI contains a description of the PM Graphic Programming Interface.	G362-0005

IBM Advanced Function Presentation Publications

Title	Order Number
Guide to Advanced Function Presentation contains a comprehensive overview of AFP™ and AFP concepts.	G544-3876
Advanced Function Presentation: Programming Guide and Line Data Reference	S544-3884
Advanced Function Presentation: Printer Information contains detailed characteristics of IBM's page printers.	G544-3290
Advanced Function Presentation: Printer Information contains detailed characteristics of IBM's legacy page printers.	G544–3290
IBM Printing Systems: Printer Information contains detailed characteristics of IBM's currently-marketed page printers.	S544–5750
Technical Reference for Code Pages	S544-3802
Technical Reference for IBM Expanded Core Fonts	S544-5228
Font Summary for the AFP Font Collection	S544-5633
IBM Advanced Function Presentation Fonts: Font Summary	G544-3810
Infoprint® Fonts: Font Summary	G544–5846
Page Printer Formatting Aid: User's Guide and Reference contains information about the PPFA product that is used to create AFP page definitions and form definitions.	S544-5284
Overlay Generation Language/370: User's Guide and Reference contains information about the OGL product that is used to create AFP overlays.	S544-3702
Advanced Function Presentation Workbench for Windows®: Using the Viewer Application contains information about using it with AFP API.	G544-3813
Advanced Function Presentation Conversion and Indexing Facility: Application Programming Guide contains information about using ACIF.	G544-3824
Advanced Function Presentation: Toolbox for Multiple Operating Systems User's Guide	G544-5292
AFP Application Programming Interface: Programming Guide and Reference contains information about using the AFP Application Programming Interface.	S544-3872
Printing and Publishing Collection Kit contains the online, softcopy version of most of the books referred to in this book.	SK2T-2921

Print Services Facility Publications

Title	Order Number
Print Services Facility/MVS: Application Programming Guide	S544-3673
Print Services Facility/VM: Application Programming Guide	S544-3677
Print Services Facility/VSE: Application Programming Guide	S544-3666
Print Services Facility/2: Getting Started	G544-3767
IBM AIX® Print Services Facility/6000: Print Services Facility™ for AIX Users	G544-3814
AS/400 Information Directory	GC21-9678

Infoprint Manager Publications

Title	Order Number
Infoprint Manager for AIX Publications (CDROM)	SK2T-9266

Changes in This Edition

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

This edition provides enhancements to IBM's bar code architecture. Changes include:

- Two new bar code types to provide additional symbol variations:
  - QR Code 2D bar code
  - Code 93 1D bar code
- Two new bar code variations:
  - PLANET, a variation of POSTNET
  - UCC/EAN 128, a variation of Code 128
- Additional information, clarifications, and pictures to aid in the generation of BCOCA objects

Contents

Preface iii
Who Should Read This Book iii
How to Use This Book iii
How to Read the Syntax Diagrams iv
Notation Conventions vi
Related Publications vii
IBM Architecture Publications vii
IBM ImagePlus Publications vii
IBM Graphics and Image Publications vii
IBM Advanced Function Presentation Publications viii
Print Services Facility Publications viii
Infoprint Manager Publications viii

Changes in This Edition ix

Figures xiii

Tables xv

Chapter 1. A Presentation Architecture Perspective 1
The Presentation Environment 1
Architecture Components 2
Data Streams 2
Objects 3
Relationship to Systems Application Architecture 5
Application Enabling Products 6

Chapter 2. Introduction to BCOCA 9
What Is a Bar Code? 9
How Data Is Presented 9
How Data Is Retrieved 9
Elements of a Bar Code System 9
Bar Code Symbology 10
Linear Symbologies 10
Two-Dimensional Matrix Symbologies 13
Two-Dimensional Stacked Symbologies 13
Bar Code Symbol Generation 14
Bar Code Encoding Techniques 14
Information Density 14
Physical Media 15
Printers 15
Scanners 16
Performance Measurement 16

Chapter 3. BCOCA Overview 17
General BCOCA Concepts 17
Bar Code Object Processor 17
Bar Code Presentation Space 19
Coordinate System 19
Measurements 19
L-unit Range Conversion Algorithm 20
Symbol Placement 21
Symbol Orientation 22
Symbol Size 23

Chapter 4. BCOCA Data Structures . . . 25
BCD1 Subset . . . 25
Bar Code Symbol Descriptor (BSD) . . . 26
Check Digit Calculation Methods . . . 59
Bar Code Symbol Data (BSA) . . . 63
Data Matrix Special-Function Parameters . . . 70
MaxiCode Special-Function Parameters . . . 75
PDF417 Special-Function Parameters . . . 81
QR Code Special-Function Parameters . . . 87
Valid Code Pages and Type Styles . . . 93
Valid Characters and Data Lengths . . . 94
Characters and Code Points (Excluding Code 128 and 2D Bar Codes) . . . 98
Code 128 Code Page . . . 100

Chapter 5. Exception Conditions . . . 101
Specification-Check Exceptions . . . 101
Data-Check Exceptions . . . 105

Chapter 6. Compliance . . . 107
Generator Rules . . . 107
Receiver Rules . . . 107
SAA Compliance . . . 107

Appendix A. Bar Code Symbology Specification References . . . 109

Appendix B. MO:DCA Environment . . . 111
Bar Codes in MO:DCA-P Documents . . . 111
Compliance with MO:DCA-P Interchange Set 2 . . . 111
Bar Code Data Object Structured Fields . . . 112
Bar Code Data Descriptor (BDD) . . . 112
Bar Code Data (BDA) . . . 112

Appendix C. IPDS Environment . . . 113
IPDS Bar Code Command Set . . . 113
Write Bar Code Control Command . . . 113
Write Bar Code Command . . . 114
Additional Related Commands . . . 114

Notices . . . 117
Trademarks . . . 119

Glossary . . . 121

Index . . . 155

Figures

1. Presentation Environment 1
2. Presentation Model 3
3. Presentation Page 5
4. Bar Code Symbol Structure 10
5. Examples of Linear Bar Code Symbols 11
6. Examples of 2D Matrix Bar Code Symbols 13
7. Example of a 2D Stacked Bar Code Symbol 13
8. Bar Code Presentation Space 19
9. Bar Code Orientations 22
10. Example of a MaxiCode Bar Code Symbol with Zipper and Contrast Block 80
11. Subset of EBCDIC Code Page 500 That Can Be Translated To GLI 0 82
12. Subset of EBCDIC Code Page 500 That Can Be Translated To ECI 000020 89
13. Code 128 Code Page (CPGID = 1303, GCSGID = 1454) 100

Tables

1. Data Structure Syntax iv
2. Field Ranges for Commonly-Supported Measurement Bases 21
3. Modifier Values by Bar Code Type 30
4. Standard OCA Color-Value Table 55
5. Supported Sizes for a Data Matrix Symbol 72
6. Supported Versions for a QR Code Symbol 90
7. Valid Code Pages and Type Styles 93
8. Valid Characters and Data Lengths. 94
9. Characters and Code Points used in the BCOCA Symbologies; Excluding Code 128 and 2D Bar Codes 98

Chapter 1. A Presentation Architecture Perspective

This chapter provides a brief overview of Presentation Architecture.

The Presentation Environment

Figure 1 shows today's presentation environment.

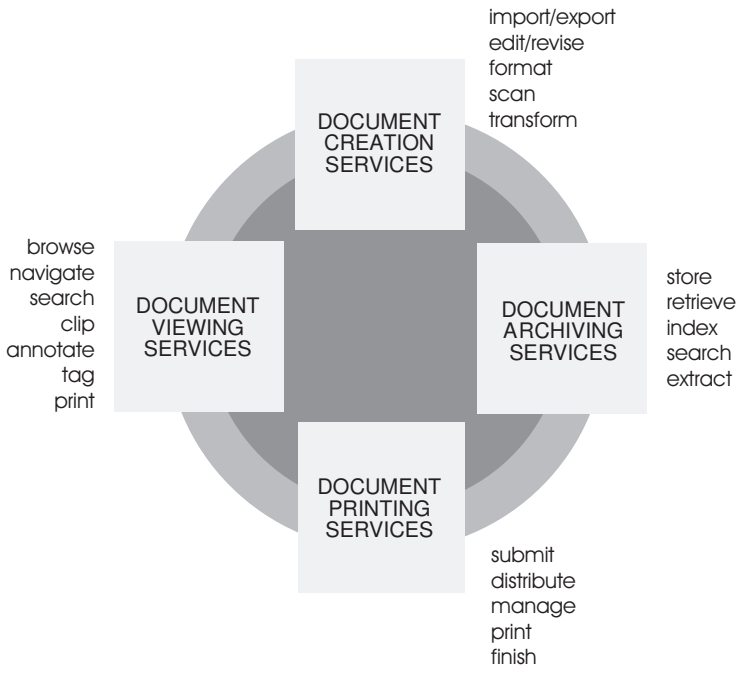


Figure 1. Presentation Environment. The environment is a coordinated set of services architected to meet the presentation needs of today's applications.

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

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

IBM presentation architectures define interchange formats that are system independent and are independent of any particular format used for physically transmitting or storing data. Where appropriate, IBM presentation architectures use

industry and international standards, such as the ITU-TSS (formerly known as CCITT) facsimile standards for compressed image data.

Architecture Components

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

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

Data Streams

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

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

The MO:DCA architecture defines the data stream used by applications to describe documents and object envelopes for interchange with other applications and application services. Documents defined in the MO:DCA format may be archived in a database, then later retrieved, viewed, annotated and printed in local or distributed systems environments. Presentation fidelity is accommodated by including resource objects in the documents that reference them.

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

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

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

Figure 2 on page 3 shows a system model relating MO:DCA and IPDS data streams to the presentation environment previously described. Also shown in the model are

the object content architectures which apply to all levels of presentation processing in a system.

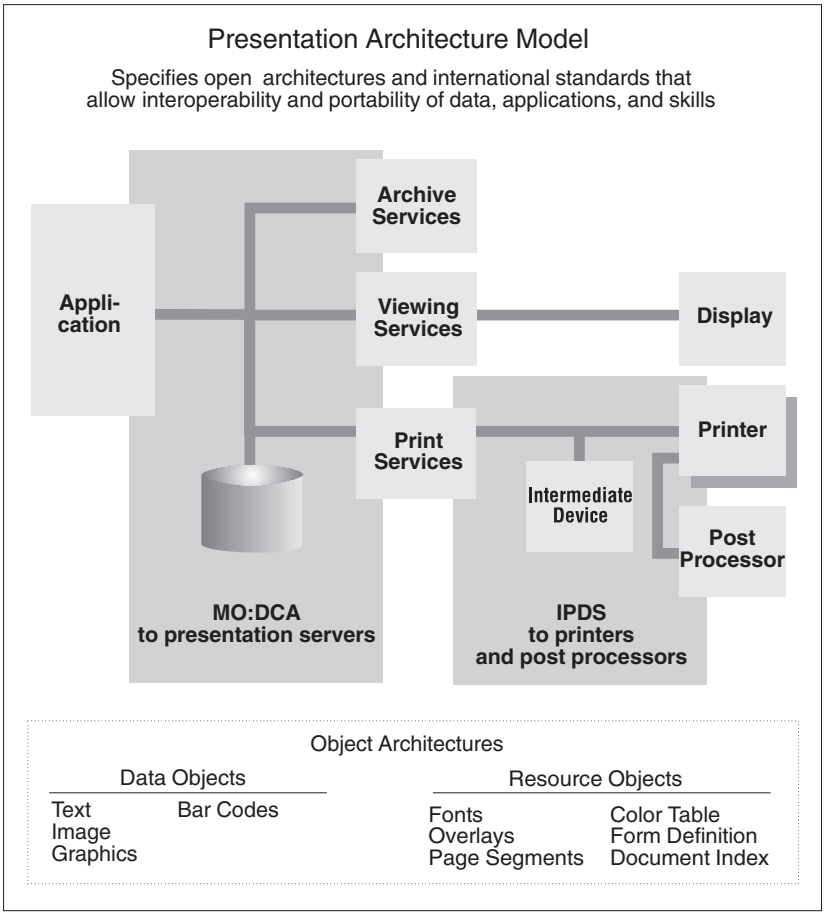


Figure 2. Presentation Model. This diagram shows the major components in a presentation system and their use of data stream and object architectures.

Objects

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

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

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

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

The object content architectures are:

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

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

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

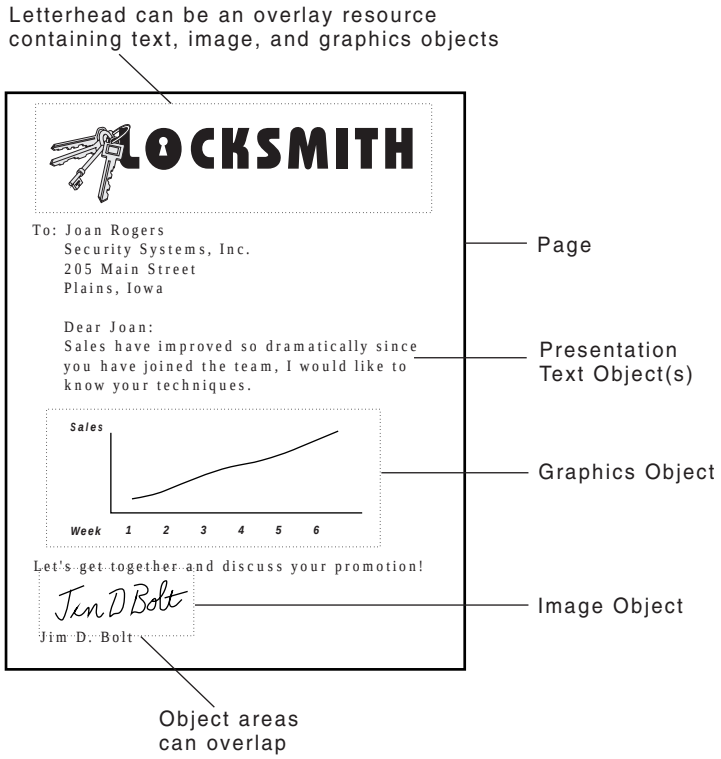


Figure 3. Presentation Page. This is an example of a mixed-object page that can be composed in a device-independent MO:DCA format and can be printed on an IPDS printer.

Relationship to Systems Application Architecture

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

Application Enabling Products

Some of the major application enabling products and application services using presentation interchange architectures are:

- *Advanced Function Presentation (AFP)*

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

- *AFP Conversion and Indexing Facility (ACIF)*

  An AFP program for converting a System/370™ line-data print file into a MO:DCA document and for indexing the document for later retrieval, viewing and selective printing of pages.

- *AFP Workbench*

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

- *AFP Toolbox*

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

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

- *Advanced Function Printing™ Utilities/400*

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

- *Graphical Data Display Manager (GDDM)*

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

- *Infoprint Manager for AIX, Windows NT®, and Windows 2000*.

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

- *OS/2 Presentation Manager GPI*

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

- *IBM SAA ImagePlus Workstation Program/2*

  An IBM licensed program designed to capture, view, annotate, print and manipulate text and image documents on an OS/2 workstation platform. Documents are generated in the MO:DCA interchange format and can be transmitted to MVS and OS/400 hosts for folder management and archival storage by other ImagePlus components.

- *IBM SAA MVS/ESA ImagePlus System*

  A set of licensed programs that are designed to work in conjunction with the ImagePlus Workstation Program/2 to provide MVS host support for Folder Applications and WorkFlow Management. Documents are stored in the MO:DCA Interchange format and are distributed on request by an Object Distribution Manager.

- *IBM SAA AS/400 ImagePlus System*

  A set of licensed programs that are designed to work in conjunction with the ImagePlus Workstation Program/2 to provide Operating System/400® (OS/400) host support for Electronic Filing Cabinets and WorkFolder applications. Documents are stored in the MO:DCA Interchange format and made available on request to workstation programs.

- *IBM SAA ImagePlus/2 System*

  A comprehensive, user-configurable, OS/2 LAN-based implementation of ImagePlus document imaging. IBM SAA ImagePlus/2 consists of two components:
  - IBM SAA ImagePlus Services Facility/2
  - IBM SAA ImagePlus Application Facility/2

  IBM SAA ImagePlus Services Facility/2 provides storage management, content class management, document, page and display management, image capture and presentation management. IBM SAA ImagePlus Application Facility/2 provides the application and end-user interface, document storage and retrieval, plus document, folder and case management. It also includes menu-driven workflow processing capabilities. Documents are stored in the MO:DCA Interchange format.

- *Print Services Facility (PSF)*

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

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

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

- *Print Services Facility/6000 (PSF/6000)*

  An AIX/6000 print server that drives IPDS page printers. In addition to managing printer resources and providing error recovery for print jobs, PSF/6000 provides data stream conversions of PostScript and ditroff data streams to MO:DCA data streams for interoperability with other AFP products on AIX/6000 and other system platforms.

For more information on these and other products, refer to the publications listed in "Related Publications" on page vii.

Chapter 2. Introduction to BCOCA

This chapter:
- Provides a brief overview of bar codes
- Describes the basic elements of a bar code system
- Describes how bar code system performance is measured

What Is a Bar Code?

A bar code is an accurate, easy, and inexpensive method of data presentation and data entry for Automatic Identification (AutoID) information systems. Bar codes are the predominant AutoID technology used to collect data about any person, place, or thing. Bar codes are used for item tracking, inventory control, time and attendance recording, check-in/check-out, order entry, document tracking, monitoring work in progress, controlling access to secure areas, shipping and receiving, warehousing, point-of sale operations, patient care, and other applications.

A bar code is a predetermined pattern of elements, such as bars, spaces, and two-dimensional modules, that represent numeric or alphanumeric information in a machine-readable form. The way the elements are arranged is called a *symbology*. The Universal Product Code (UPC), the European Article-Numbering (EAN) system, Code 39, Interleaved 2-of-5, and Code 128 are some examples of symbologies.

How Data Is Presented

Physical media and printers are used to present bar code data. Paper is the most common form of physical media used to present data—for example, retail shelf labels, shipping containers, books, documents, electronic forms, and mailing envelopes. However, other physical media are also used, such as fabric labels and corrosive-resistant metal tags. The physical media must be durable enough to withstand the expected wear and have the requisite optical properties to allow scanning equipment to read the bar code successfully. Symbol printing can occur either on-demand in real-time or off-line in a batch printing process. The printer technology, printer element size, printer tolerances, and optical properties of the physical media and marking agent all determine the readability of the bar code.

How Data Is Retrieved

Data contained in a bar code symbol is retrieved by scanning the printed elements with an optical device called a *scanner*. The scanning device develops logic signals corresponding to the difference in reflectivity of the printed bars and the underlying physical media. The logic signals are translated from a serial pulse stream into digitized computer readable data by a device called a *decoder*. The digitized data is transmitted to the host computer for processing.

Elements of a Bar Code System

A bar code system consists of four major elements:
1. The bar code symbology used to encode the data
2. The physical media on which the bar code is printed
3. The type of printing device used to print the bar code on the physical media
4. The scanning device used to read the bar code.

The following sections describe these elements in greater detail.

Bar Code Symbology

Linear Symbologies

A bar code symbol consists of six parts, as illustrated in Figure 4. The complete symbol consists of a start margin, a start character, the data or message characters, an optional check-digit character, a stop character, and a stop margin.

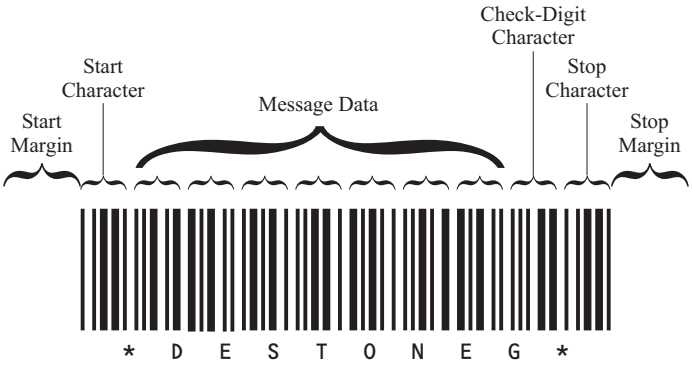


Figure 4. Bar Code Symbol Structure

The *start and stop margins*, sometimes referred to as *quiet zones*, are void of any printed character. They are typically white. The margin areas are used to instruct the decoder that the scanner is about to encounter a bar code symbol.

The *start character*, which precedes the first character of the bar code message, is a special bar and space pattern used to identify the beginning of a bar code symbol. The start character enables the decoder to determine that a bar code symbol is being scanned and not some other sequence of reflective and non-reflective areas which may have the same pattern as one of the characters in the symbol.

The *message* portion of the symbol contains the data to be stored. The data characters are encoded as a series of parallel bar and space patterns according to the bar code symbology used. Refer to Appendix A, "Bar Code Symbology Specification References," on page 109 for a list of the bar code symbology specifications.

Most bar code symbologies define a mandatory or optional *check-digit character* (or characters). The value of the check-digit character is determined by an arithmetic operation performed on the data characters in the message when the symbol is created. When used, the check-digit character becomes the last character of the message immediately preceding the stop character.

The *stop character* is also a special bar and space pattern. Its purpose is to signal the end of the symbol. When a check-digit character is used, the stop character instructs the decoder to perform the check-digit calculation on the message data characters and compare the computed value to the encoded check-digit character. The decoder also uses the stop character to know that it may decode and validate the message data characters. If the message data characters are valid, the data is transmitted to the host computer for processing. Otherwise, an error signal is generated.

The bar and space patterns used to encode the start and stop characters are generally not symmetrical, that is, the same bar and space pattern is not used for both characters. This feature enables a decoder to scan in the forward or reverse directions.

Figure 5 shows examples of linear bar code symbols.

Australia Post Bar Code
Customer Barcode 2 using Table C
(encoding 56439111ABA 9)

Japan Postal Bar Code
Modifier X'00'
(encoding 15400233-16-4)

US POSTNET
Zip+4
(encoding 12345+6789)

PLANET Code
(encoding 00123456789)

Royal Mail (RM4SCC)
UK and Singapore version
(encoding SN34RD1A)

Royal Mail (RM4SCC)
Dutch KIX version
(encoding SN34RD1A)

Code 39 (3-of-9 Code)
(encoding 39OR93 with check character
yielding a 2.32 inch wide symbol)

Code 93
(encoding 39OR93
yielding a 1.82 inch wide symbol)

Figure 5. Examples of Linear Bar Code Symbols (Part 1 of 2)

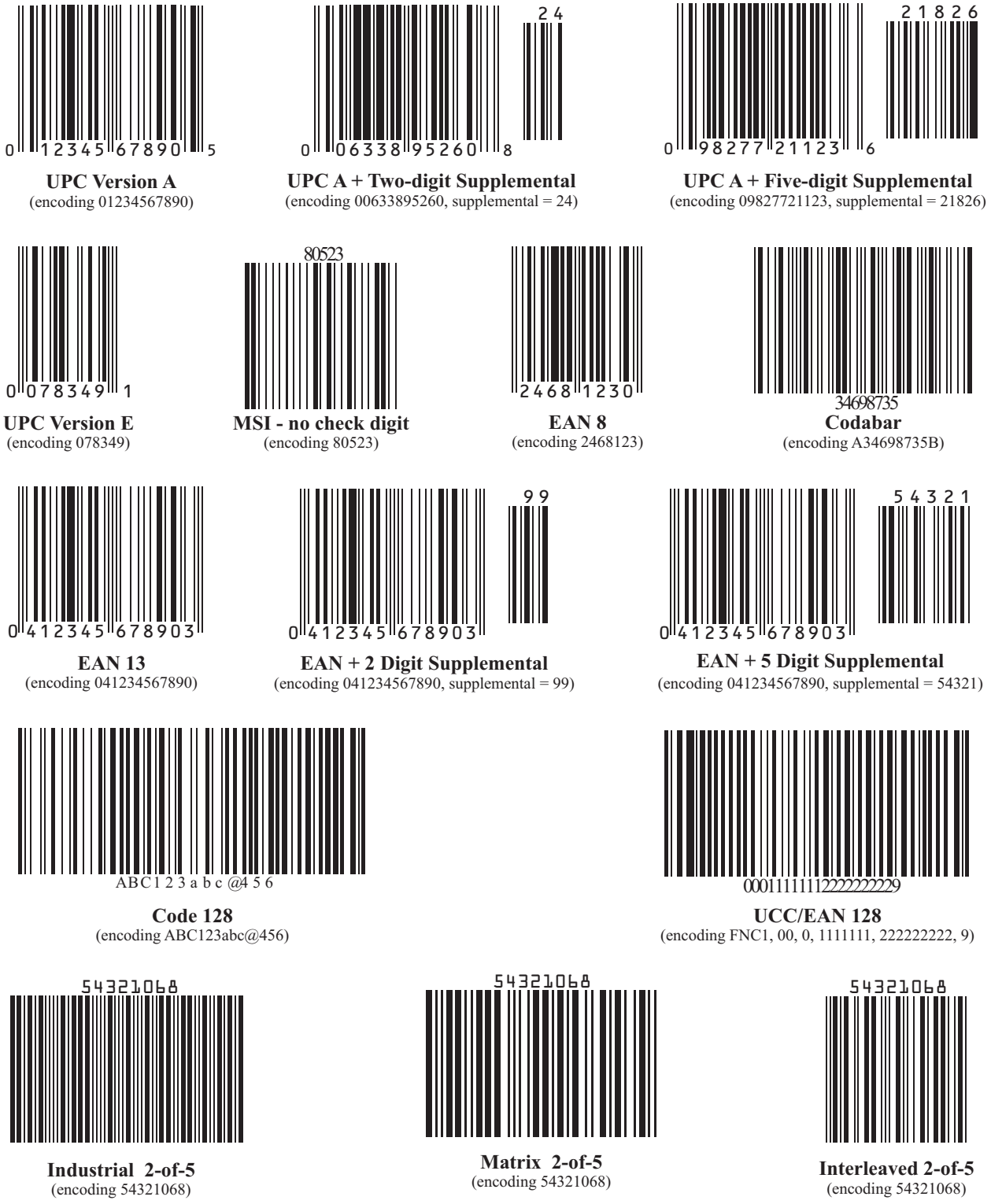


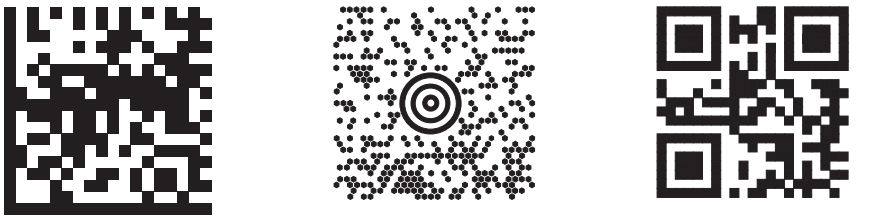
Figure 5. Examples of Linear Bar Code Symbols (Part 2 of 2)

Two-Dimensional Matrix Symbologies

Two-dimensional matrix symbologies (sometimes called area symbologies) allow large amounts of information to be encoded in a two-dimensional matrix. These symbologies are usually rectangular and require a quiet zone around all four sides; for example, the Data Matrix symbology requires a quiet zone at least one module wide around the symbol. Two-dimensional matrix symbologies use extensive data compaction and error correction codes, allowing large amounts of character or binary data to be encoded.

Unlike linear bar codes, Human-Readable Interpretation (HRI) is not provided with the bar code symbol.

Figure 6 shows examples of two-dimensional matrix bar code symbols.



Data Matrix 2D Symbol
(encoding A1B2C3D4E5F6G7H8I9J0K1L2)

MaxiCode 2D Symbol

QR Code 2D Symbol

Figure 6. Examples of 2D Matrix Bar Code Symbols

Two-Dimensional Stacked Symbologies

Two-dimensional stacked symbologies allow large amounts of information to be encoded by effectively stacking short one-dimensional symbols in a row/column arrangement. This reduces the amount of space that is typically consumed by conventional linear bar code symbols and allows for a large variety of rectangular bar code shapes. Figure 7 shows an example of a two-dimensional stacked symbology.

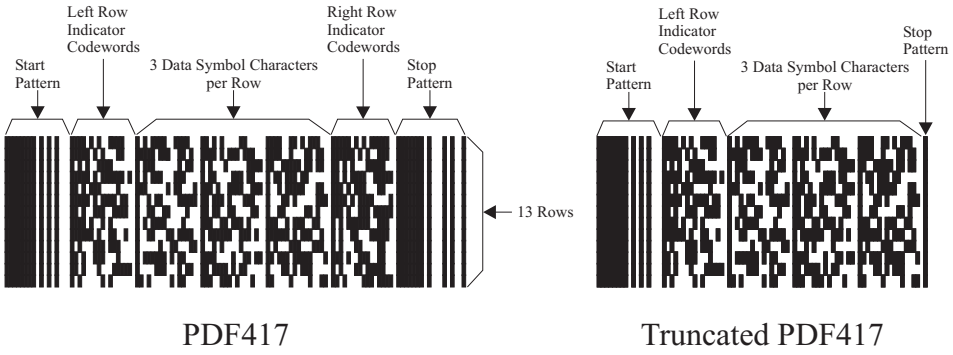


Figure 7. Example of a 2D Stacked Bar Code Symbol

Bar Code Symbol Generation

Generating a bar code symbol is a four-step process:

1. Identify the bar code symbology to be used and the data to be encoded in the message.
2. Translate the data characters into a binary sequence for encoding.
3. Create the bar and space pattern that represents each character.
4. Format the individual characters into a completed bar code symbol.

The general structure of a bar code symbol is implemented differently in each of the bar codes symbologies. The various symbologies can be categorized according to the encoding technique used and the information density.

Bar Code Encoding Techniques

There are two commonly used encoding techniques: *module width* and *non-return-to-zero* (NRZ) encoding. Module width encoding techniques are generally used in industrial applications. Commercial applications generally use NRZ. Data in module width encoding is represented differently from data in NRZ encoding.

Module width encoding techniques encode binary data through the contrast of wide and narrow element widths. A narrow element (bar or space) is known as the *module width* and represents data whose logic value is zero. A wide element (bar or space) represents data whose logic value is one and whose width is typically two to three times the narrow element. The ratio of elements or *wide-to-narrow ratio* (WE:NE) is one of the distinguishing features of the symbologies using this technique. These bar codes are referred to as *two-level codes*. With this technique, there are definite transitions from black to white and white to black separating each binary bit from its adjacent binary bits. Examples of bar code symbologies that use this form of encoding are Code 39 and Interleaved 2-of-5.

NRZ encoding techniques encode binary data through the reflectivity of the bars and spaces. A logic value of zero is represented as a reflective surface and the logic value of one as a non-reflective surface. There is no transition between bits unless the logic state changes. Therefore, a sequence of logic zeros and ones may be represented by the width of a single reflective or non-reflective element. Bar codes utilizing NRZ encoding techniques are sometimes referred to as *four-level codes* because up to four data bits of the same logic value may be contained within a single reflective or non-reflective element. Examples of bar code symbologies that use this form of encoding are UPC and EAN.

Information Density

Information density is the number of message characters that can be encoded per unit length. Density is commonly divided into three categories: high, medium, and low. A high-density bar code generally contains more than eight characters per inch; a medium density bar code contains from four to eight characters per inch; a low density bar code contains less than four characters per inch.

Two factors influence bar code density: the code structure (two-level or four-level) and the module width. Bar code density increases or decreases by varying the module width when it is printed. Module widths are generally separated into three groups: high resolution, medium resolution, and low resolution. High-resolution module widths are typically less than 0.009 inch; medium resolution module widths are between 0.009 inch and 0.020 inch; low resolution module widths are greater than 0.020 inch. The criteria for selecting module widths are the application requirements and the printer characteristics.

Physical Media

Bar code symbols can be printed on a wide variety of physical media. The most common physical media are adhesive labels, cards, and documents. Since the physical media functions as an optical storage device, the optical characteristics are very important. Specifically, the surface reflectivity of the physical media at a specific optical wavelength and the radiation pattern are critical.

Surface reflectivity is defined by the amount of light reflected when an optical emitter irradiates the physical media surface. As a general industry guideline, the physical media should reflect between 70% and 90% of the incident light. A white physical media is generally used to achieve this high reflectivity. The reflected radiation pattern is defined in terms of how the optical pattern leaves the physical media. A shiny surface results in a narrow radiation pattern. A dull or matte surface produces a diffused, or broad, pattern. Narrow radiation patterns may cause problems for scanners.

Another optical characteristic is the transparency of the physical media. If the physical media is too transparent, the material underneath the label, card, or document affects the reflectivity. Paper bleed occurs with transparent or translucent physical media. Paper bleed is caused by the scattering of incident light rays within the physical media or from the underlying surface. This scattered light is picked up by the scanner adding to the reflecting light off the physical media surface and increases the reflected signal. The result tends to make the bars appear larger and the spaces appear narrower than what was actually printed.

Printers

A wide variety of printers can print bar codes. Both impact and non-impact printers are used to achieve low, moderate, or high speed throughput. The types of printing technologies include — drum, daisywheel, dot matrix, thermal, thermal transfer, ink jet, laser, electrostatic, letterpress, lithography, offset, gravure, and flexography. The drum, dot matrix, thermal, and daisywheel printing systems are used for low to moderate throughput applications. Ink jet, laser, electrostatic, and others, are used for high throughput. Regardless of the printing technology used, print quality is the critical factor in producing machine readable bar code symbols.

Print quality is determined by the print mechanism, the physical media, and the marking agent. The major factors influencing print quality are:

- Marking agent spread/shrink
- Marking agent voids/specks
- Marking agent smearing
- Marking agent non-uniformity
- Bar and space width tolerances
- Bar edge roughness

All of these factors are potential sources of system errors. They must be closely controlled to ensure readable bar code symbols.

Scanners

Data stored in a bar code symbol is retrieved by the movement of an optical scanner across the symbol, or vice versa. The scanner may be statically mounted, as in a conveyor system, or movable, as with a hand-held wand. The scanner functions are the same.

Binary data encoded in the wide or narrow bars and spaces is extracted by the scanner's optical system. The optical system consists of an emitter, a photodetector, and an optical lens. The emitter sends a beam of light through the optical lens over the symbol, while the photodetector simultaneously responds to changes in the reflected light levels. The photodetector produces a high output current when the reflected signal is large and a low output current when the reflected signal is small. A low reflected signal occurs when the beam is over a bar. Conversely, a high reflected signal occurs when the beam is over a space. These changes in current result in an analog waveform. The waveform is processed by the decoder, which digitizes the information. The digitized information is then sent to the host computer for processing.

Performance Measurement

The performance of bar code systems is generally described in terms of two parameters. The first parameter is called the *first read rate*. The term is defined as the ratio of the number of good scans, or reads, to the number of scan attempts. Typically, a good bar code system should have a first read rate of better than 80%. A low first read rate is normally caused by a poorly printed symbol.

The second parameter used to evaluate system performance is the *substitution error rate*. This is the ratio of the number of invalid, or incorrect, characters entered into the data base to the number of valid characters entered. Substitution error rate is dependent on the structure of the bar code symbology, the quality of the printed symbol, and the design of the decoding algorithm.

Chapter 3. BCOCA Overview

This chapter provides an overview of the BCOCA architecture and describes:
- General BCOCA concepts
- Bar code object processor concepts
- Bar code presentation space concepts

General BCOCA Concepts

The BCOCA architecture is an object content architecture used to describe and generate bar code symbols.

BCOCA objects can exist in, or be invoked by, a number of environments. Each of these controlling environments can be specialized for a particular application area. For example, the controlling environment can be:

- The environment involved in electronically distributing documents in a network, for example, the MO:DCA environment
- A presentation system communicating with hard-copy presentation devices, for example, the IPDS environment

In these environments, multiple bar code symbols with the same attributes can be specified within a single bar code object as described in Appendix B, "MO:DCA Environment," on page 111 and Appendix C, "IPDS Environment," on page 113. When multiple bar code symbols of the same type are to be printed on a page, better performance can be achieved by combining the bar codes into a single object rather than having multiple bar code objects on the page.

Bar Code Object Processor

A BCOCA receiver consists of a bar code object processor. The primary function of the bar code object processor is to develop one or more bar code symbols of the same type within an abstract presentation space, as illustrated in Figure 8 on page 19. In turn, these abstract bar code presentation spaces are mapped into areas defined within the controlling environments. Examples of controlling environment areas include the IPDS bar code object area for printing bar code symbols, and the MO:DCA object area for interchange. For additional information, refer to Appendix B, "MO:DCA Environment," on page 111 and Appendix C, "IPDS Environment," on page 113.

Input to the bar code object processor consists of:
- Data to be encoded
- Bar code symbology to be used
- Bar code presentation space size parameters
- Bar code symbol location within the bar code presentation space
- Module width of the narrow bar code element
- Total element height of the bar code symbol
- Check digit generation option
- Wide-to-narrow element ratio
- Human-readable interpretation (HRI) presence, location, and type style
- Color of the bar code symbol elements
- For 2D symbologies, special functions such as:
  - Ability to ignore escape sequences

- Application indicator
- EBCDIC-to-ASCII translation
- Error correction level
- Macro characters to indicate a specific header or trailer
- Matrix row size
- Number of data symbol characters per row
- Number of rows
- Security level
- Structured append information
- Symbol conforms to specific industry standards
- Symbol is reader programming information
- Symbol mode
- Test pattern (zipper)
- Version

The bar code object processor:

- Validates all input parameters and generates exception conditions as appropriate.
- Generates the bar and space patterns of the input data to be encoded according to the rules of the specified bar code symbology.
  - For two-level codes, the bar and space patterns are generated using the module width and wide-to-narrow ratio input parameters.
  - For discrete codes, whose bar and space patterns for each character start and end with a bar, an intercharacter gap is required. The bar code object processor automatically inserts these gaps. The intercharacter gap is one module width wide.
- Generates, uses, and encodes check digit(s) according to the rules of the symbology and the check-digit option input parameter (modifier field).
- For 2D matrix symbologies, encodes and compacts the data, inserts codewords for special functions, generates ECC characters, determines the proper placement of the bits in the matrix, and generates the finder patterns.
- For 2D stacked symbologies, generates codewords from the input data using a combination of compaction schemes based on the input data, generates start and stop patterns, generates the left row and right row indicator codewords (which have the number of rows and columns and security level encoded within), generates the symbol length descriptor, and generates the error correction and detection codewords.
- Generates the appropriate start and stop bar and space patterns for all bar code types and versions including the UPC-family center and delineator patterns.
- Generates the HRI text characters and places them above or below the symbol as directed.
- Suppresses presentation of the bar code symbol if directed by the *suppress bar code symbol* flag. This can be used to print just the HRI.
- Places the bar code symbol and HRI, if present, in the bar code presentation space at the location specified. The user is responsible for insuring that the symbol quiet zones and HRI information are totally contained within the bar code presentation space.

Notes:

1. The BCOCA object generator is responsible for insuring that the bar code presentation space is large enough to allow for appropriate quiet zones. Some symbologies require extra space if a wand-type scanner is to be used.

2. All bar code symbols must be presented in their entirety. Whenever a partial bar code pattern is presented, for whatever reason, it is obscured to make it unscannable.

Bar Code Presentation Space

A bar code presentation space is a linear, two-dimensional space. An orthogonal coordinate system is used to define any point within the presentation space. Distances within the coordinate system are measured in *logical units*, also known as *L-units*. One or more bar code symbols of the same type may be placed within the presentation space. Figure 8 shows a bar code presentation space containing two bar code symbols.

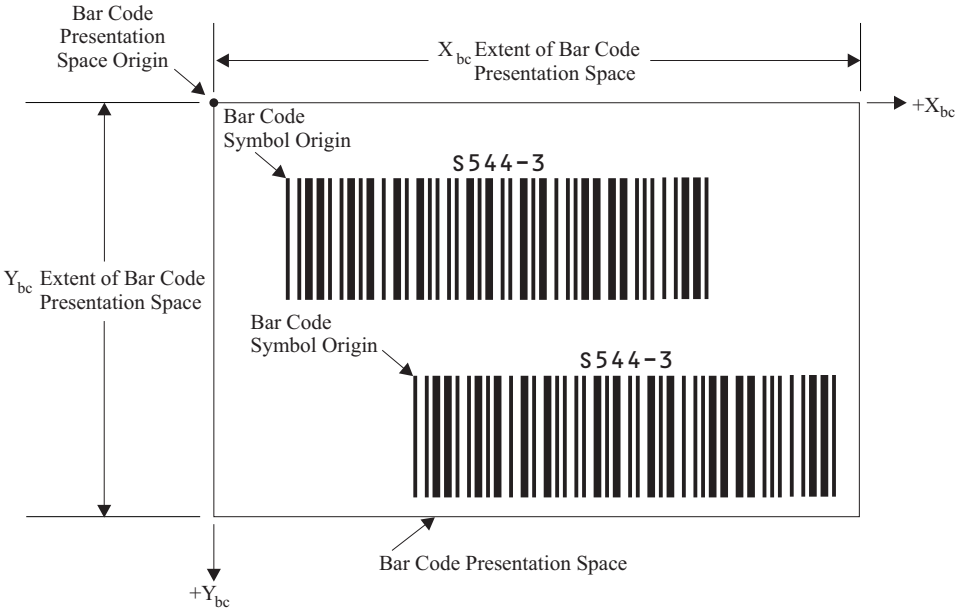


Figure 8. Bar Code Presentation Space

Coordinate System

The X_{bc},Y_{bc} coordinate system is the bar code presentation space coordinate system. The origin of this system (x_{bc}=0, y_{bc}=0) is the top-left corner. Positive X_{bc} values increase from left to right. Positive Y_{bc} values increase from top to bottom.

The size of the bar code presentation space in the X_{bc} dimension is called the X_{bc} *extent*. The size of the bar code presentation space in the Y_{bc} dimension is called the Y_{bc} *extent*.

Measurements

In general usage, linear measurements are expressed as a specific number followed by a unit called the *measurement base*. The measurement base is typically a well known unit such as an inch or a centimeter. For example, in the measurement *12 inches*, the measurement base is *inches*; in the measurement *12 centimeters*, the measurement base is *centimeters*. Since we know the length of one inch or one centimeter, it is easy to measure 12 of these units.

In BCOCA data structures, linear measurements are expressed as numbers called *logical units (L-units)*. When a number is expressed in terms of L-units, an

appropriate measurement base must be used to interpret the value of the number. The measurement base is separately supplied in the Bar Code Symbol Descriptor (BSD).

Measurement bases used in BCOCA objects are expressed using a *unit base* field and a *units per unit base* field:

Unit base — A one-byte code that represents the length of the measurement base. A value of X'00' specifies that the length of the measurement base is ten inches. A value of X'01' specifies that the length of the measurement base is ten centimeters.

Units per unit base — A two-byte field that contains the number of units in the measurement base. The previous general-usage examples had a unit base of one inch or one centimeter and a units per unit base of one. The BCOCA architecture allows the units per unit base to be any value between X'0001' and X'7FFF', but requires all bar code object processors to at least support X'3840' (14400) units per ten inches. Many bar code object processors also support X'0960' (2400) units per ten inches.

For example, within bar code symbol data, the X and Y offset values for placing the bar code symbol within the presentation space might be expressed as X'00F0' (240) L-units in the X-direction and X'01E0' (480) L-units in the Y-direction. For a unit base of X'00' (ten inches) with 2400 units per unit base, this describes a point 1 inch over and 2 inches down from the origin of the presentation space.

Units of measure is the length of the measurement base, specified by the unit base field, divided by the value of units per unit base. For example, the units of measure for a bar code presentation space might be expressed as 1/240 of an inch; there are 240 units in one inch. The term *L-unit* is sometimes used as a synonym for unit of measure.

Resolution is the reciprocal of units of measure. For example, the resolution of the bar code presentation space would be expressed as 240 units per inch.

L-unit Range Conversion Algorithm

Some field values within BCOCA data structures are specified assuming a unit of measure of 1/1440 of an inch. These fields are designated as such with a reference to this algorithm. If a BCOCA receiver supports additional units of measure, the BCOCA architecture requires the receiver to at least support a range equivalent to the specified range relative to each supported unit of measure. Table 2 on page 21 lists the equivalent field ranges for the most commonly used units of measure.

The values required to be supported when 14400 units per 10 inches is specified for a field are listed in the BCOCA data structure. If additional units of measure are supported, the field values that the BCOCA architecture requires a bar code object processor to support for these alternate units of measure are calculated using the following algorithm:

1. Calculate the number of supported units per inch as follows:
   - If the length of the measurement base for a field is ten inches, divide the number of supported units that applies to the desired field by ten.

- If the length of the measurement base for a field is ten centimeters, multiply the number of supported units per ten centimeters (one decimeter) that applies to the desired field by 0.254, the approximate number of decimeters per inch.

2. Calculate the number of supported units per BCOCA unit as follows:
   - Divide the number of supported units per inch calculated in the previous step by 1440 (the number of BCOCA units per inch).
3. Calculate the required value in the supported unit of measure as follows:
   - Multiply the BCOCA-specified subset range values for the desired field, after converting to base ten, by the supported units per BCOCA-specified unit calculated in the previous step.
   - Round off the product to the nearest integer; for example, 2.5 would become 3 and 2.4 would become 2.
   - Adjust the new range so that it is a subset of the BCOCA-specified subset range.

For example, suppose that the specified range is X'0001'–X'7FFF' when using 14400 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 = 2400 / 10 = 240
2. Supported units per BCOCA unit = 240 / 1440 = 1/6
3. Range at 2400 units per 10 inches:
   a. X'0001' = 1 (converted to base ten)
      (1)(1/6) = 0.1667
   b. X'7FFF' = 32767 (converted to base ten)
      (32767)(1/6) = 5461.1667

Therefore, the equivalent range at 2400 units per 10 inches is “1 to 5461” which in hexadecimal is X'0001' to X'1555'. Table 2 shows the BCOCA-required ranges for several commonly supported measurement bases.

Table 2. Field Ranges for Commonly-Supported Measurement Bases

14400 units per 10 inches	**5670 units per 10 centimeters**	**2400 units per 10 inches**	**945 units per 10 centimeters**
X'0001'–X'7FFF'	X'0001'–X'7FFF'	X'0001'–X'1555'	X'0001'–X'1555'

Symbol Placement

One or more bar code symbols may be placed within the bar code presentation space. The origin of the bar code symbol is defined to be the top, left-hand corner of the left-most bar of the symbol. The height of the symbol is measured in the $+Y_{bc}$ direction. The width of the symbol is measured in the $+X_{bc}$ direction.

The BCOCA object generator is responsible for insuring that the bar code presentation space is large enough to allow for appropriate quiet zones. Some symbologies require extra space if a wand-type scanner is to be used. Exception condition EC-1100 exists if any portion of the bar code, including the bar and space patterns and the HRI, extends outside of the bar code presentation space.

Symbol Orientation

Orientation of a bar code symbol into either the *picket fence* or *ladder* orientation is accomplished by rotating the bar code presentation space within the controlling environment. In the MO:DCA environment this orientation is specified in the Object Area Position (OBP) structured field; in the IPDS environment this orientation is specified in the Bar Code Area Position (BCAP) structure in the Write Bar Code Control (WBCC) command.

A picket fence bar code or symbol is presented horizontally. In this orientation, the bars look like a picket fence. A ladder bar code or symbol is presented vertically. In this orientation, the bars look like the rungs of a ladder. Figure 9 shows two different bar code symbols as examples of the two orientations.

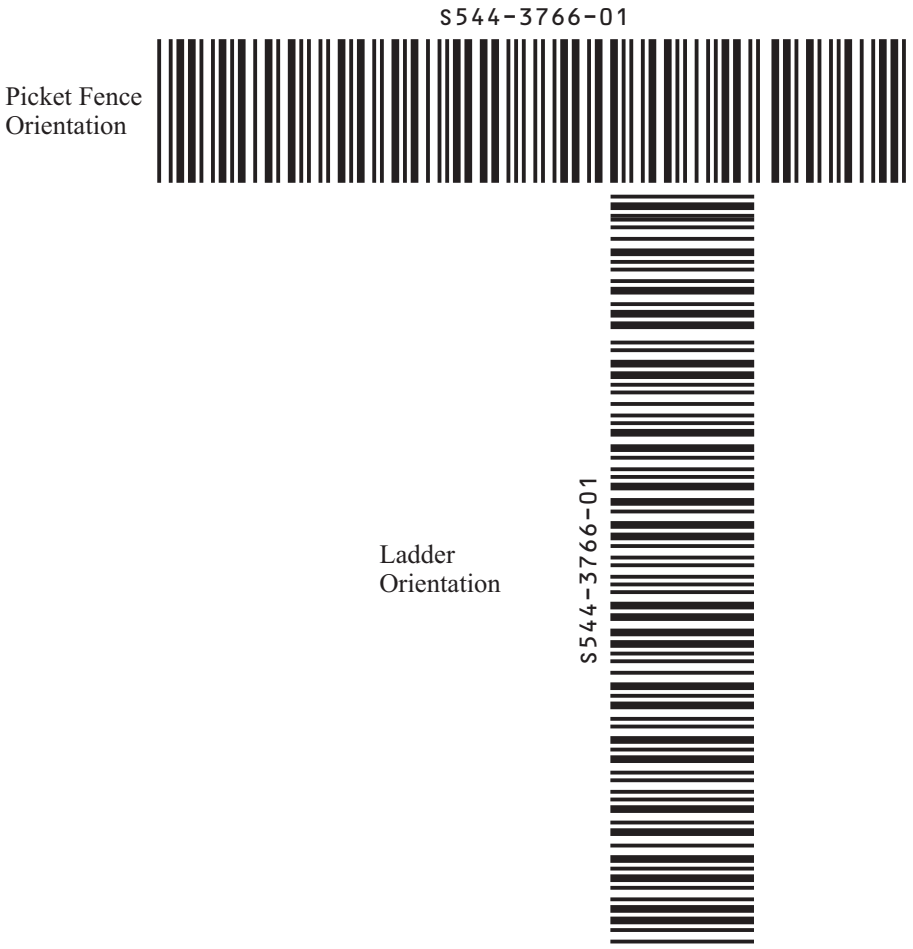


Figure 9. Bar Code Orientations

Symbol Size

The height of a bar code symbol is controlled by the bar code symbology definition, by the amount of data to be encoded, and by various BCOCA parameters. The width of the symbol is usually dependent on the amount of data to be encoded and by choices made in various BCOCA parameters. Default values exist for most of the BCOCA parameters that can be used to produce minimal-size, scannable symbols; refer to your printer documentation for information about the specific default values used by BCOCA printers.

Linear Symbologies

The element-height and height-multiplier parameters specify the height of the symbol. For some bar code types, these parameters also include the height of the human-readable interpretation (HRI). Refer to the description of the element-height parameter on page 57 for a description of the height for specific linear symbols. Some bar code symbologies (Australia Post Bar Code, Japan Postal Bar Code, POSTNET, and RM4SCC) explicitly specify the bar code symbol height; in this case, the element-height and height-multiplier parameters are ignored.

Two-Dimensional Matrix Symbologies

The MaxiCode symbology specifies a fixed physical size, nominally 28.14 mm wide by 26.91 mm high; the module-width, element-height, and height-multiplier parameters are ignored for MaxiCode symbols.

Data Matrix symbols are rectangular and are made up of a pattern of light and dark squares (called modules). The size of each module is specified in the module-width parameter and the number of rows and columns of these modules is controlled by the desired-number-of-rows and desired-row-size parameters and the amount of data to be encoded. The element-height and height-multiplier parameters are ignored for Data Matrix symbols.

QR Code symbols are square and are made up of a pattern of light and dark squares (called modules). The size of each module is specified in the module-width parameter; the number of rows and columns of these modules is controlled by the version parameter, the error correction level selected, and the amount of data to be encoded. The element-height and height-multiplier parameters are ignored for QR Code symbols.

Two-Dimensional Stacked Symbologies

PDF417 symbols are rectangular and are made up of a pattern of light and dark rectangles (called modules). The size of each module is specified in the module-width, element-height, and height-multiplier parameters and the number of rows and columns of these modules is controlled by the data-symbols and rows parameters and the amount of data to be encoded. A PDF417 symbol must contain at least 3 rows.

Chapter 4. BCOCA Data Structures

This chapter contains the BCOCA data structures, fields, and valid data definitions. Two data structures are described: the Bar Code Symbol Descriptor (BSD) and the Bar Code Symbol Data (BSA).

BCD1 Subset

The BCOCA architecture provides a wide range of bar code function to cover many different symbologies that are defined for a variety of uses. Not all of the defined BCOCA function is supported by all BCOCA receivers.

A subset of the full capabilities of the BCOCA architecture, called BCD1, is defined to specify the minimum support required of all BCOCA receivers. Each field within a BCOCA data structure allows a range of possible values which is shown in the *Range* column of the syntax table; the *BCD1 Range* column specifies the values that every receiver supports. Most receivers support more than the minimum ranges.

Bar Code Symbol Descriptor (BSD)

The BSD specifies the size of the bar code presentation space, the type of bar code to be generated, and the parameters used to generate the bar code symbols.

Offset	Type	Name	Range	Meaning	BCD1 Range
0	CODE	UBASE	X'00' X'01'	Ten inches Ten centimeters	X'00'
1			X'00'	Reserved	
2–3	UBIN	XUPUB	X'0001'–X'7FFF'	Units per unit base in the X_{bc} direction	X'3840'
4–5	UBIN	YUPUB	X'0001'–X'7FFF'	Units per unit base in the Y_{bc} direction; must be the same as XUPUB	X'3840'
6–7	UBIN	XEXTENT	X'0001'–X'7FFF' X'FFFF'	Width of bar code presentation space in L-units Default	X'0001'–X'7FFF' (Refer to the note following the table.) X'FFFF'
8–9	UBIN	YEXTENT	X'0001'–X'7FFF' X'FFFF'	Length of bar code presentation space in L-units Default	X'0001'–X'7FFF' (Refer to the note following the table.) X'FFFF'
10–11			X'0000'	Reserved	
12	CODE	TYPE	X'01'–X'03', X'05'–X'0D', X'11', X'16'–X'18', X'1A'–X'21'	Bar code type	X'01'–X'03', X'05'–X'09', X'0C', X'16'–X'17'
13	CODE	MOD	See field description	Bar code modifier	Specified in Table 3 on page 30.
14	CODE	LID	X'00'–X'FE' X'FF'	Font Local ID for HRI Default	X'01'–X'7F' X'FF'
15–16	CODE	COLOR	X'0000'–X'0010', X'FF00'–X'FF08', X'FFFF'	Color	X'FF07'
17	UBIN	MODULE WIDTH	X'01'–X'FE' X'FF'	Module width in mils Default	Receiver specific X'FF'
18–19	UBIN	ELEMENT HEIGHT	X'0001'–X'7FFF' X'FFFF'	Element height in L-units Default	X'0001'–X'7FFF' (Refer to the note following the table.) X'FFFF'
20	UBIN	MULT	X'01'–X'FF'	Height multiplier	X'01'–X'FF'
21–22	UBIN	WE:NE	X'0000' X'0001'–X'7FFF' X'FFFF'	Bar code (see byte 12) does not use wide-to-narrow ratio. Wide-to-narrow ratio. Default	X'0000' At least one value X'FFFF'

Note: The BCD1 range for these fields have been specified assuming a unit of measure of 1/1440 of an inch. Many receivers support the BCD1 subset plus additional function. If a receiver supports additional units of measure, the BCOCA architecture requires the receiver to at least support a range

equivalent to the BCD1 range relative to each supported unit of measure. More information about supported-range requirements is provided in the section titled "L-unit Range Conversion Algorithm" on page 20.

The following is a description of the fields defined in the BSD data structure and applicable exception conditions. Unless explicitly specified, the standard action to be taken for all exception conditions is to report the exception condition, terminate the bar code object processing, and continue processing with the next object.

Byte 0 UBASE

Indicates the length of the measurement unit base. The value X'00' indicates that the measurement unit base is ten inches. The value X'01' indicates that the measurement unit base is ten centimeters. Exception condition EC-0505 exists if the unit base specified is invalid or unsupported.

Byte 1 Reserved

Bytes 2–3 XUPUB

Specifies the number of units per unit base in the X_{bc} direction. Exception condition EC-0605 exists if the units per unit base value specified is invalid or unsupported.

Bytes 4–5 YUPUB

Specifies the number of units per unit base in the Y_{bc} direction and must be equal to the value specified in XUPUB. Exception condition EC-0605 exists if the units per unit base value specified is invalid or unsupported.

Bytes 6–7 XEXTENT

Specifies the width in the X_{bc} direction of the presentation space in L-units. The measurement base is specified in bytes 0–5. A value of X'FFFF' indicates that the width of the controlling environment area in the X_{bc} direction is to be used. Exception condition EC-0705 exists if the presentation space extent specified is invalid or unsupported.

Note: The size of a bar code symbol is not always known in advance. It is good practice to specify the size of the bar code presentation space large enough to include plenty of white space around the expected symbols and HRI.

Bytes 8–9 YEXTENT

Specifies the length in the Y_{bc} direction of the presentation space in L-units. The measurement base is specified in bytes 0–5. A value of X'FFFF' indicates that the length of the controlling environment area in the Y_{bc} direction is to be used. Exception condition EC-0705 exists if the presentation space extent specified is invalid or unsupported.

Bytes 10–11 Reserved

Byte 12 TYPE

Indicates the type of bar code symbol to be generated. Exception condition EC-0300 exists if the bar code type value is invalid or unsupported. Exception condition EC-1100 exists if a portion of the bar code symbol extends beyond the intersection of the mapped bar code presentation space and the controlling environment object area or beyond the maximum presentation area.

The bar code types are defined as follows:

Code	Bar Code Type	In BCD1 Subset?
X'01'	Code 39 (3-of-9 Code), AIM USS-39	Yes
X'02'	MSI (modified Plessey code)	Yes
X'03'	UPC/CGPC—Version A	Yes
X'05'	UPC/CGPC—Version E	Yes
X'06'	UPC—Two-digit Supplemental (Periodicals)	Yes
X'07'	UPC—Five-digit Supplemental (Paperbacks)	Yes
X'08'	EAN-8 (includes JAN-short)	Yes
X'09'	EAN-13 (includes JAN-standard)	Yes
X'0A'	Industrial 2-of-5	No
X'0B'	Matrix 2-of-5	No
X'0C'	Interleaved 2-of-5, AIM USS-I 2/5	Yes
X'0D'	Codabar, 2-of-7, AIM USS-Codabar	No
X'11'	Code 128, AIM USS-128	No
X'16'	EAN Two-digit Supplemental	Yes
X'17'	EAN Five-digit Supplemental	Yes
X'18'	POSTNET	No
X'1A'	RM4SCC	No
X'1B'	Japan Postal Bar Code	No
X'1C'	Data Matrix (2D bar code)	No
X'1D'	MaxiCode (2D bar code)	No
X'1E'	PDF417 (2D bar code)	No
X'1F'	Australia Post Bar Code	No
X'20'	QR Code (2D bar code)	No
X'21'	Code 93	No

Abbreviations used in this book have the following meanings:

AIM USS	Automatic Identification Manufacturers Uniform Symbol Specification
EAN	European Article Numbering
JAN	Japanese Article Numbering
MSI	MSI Data Corporation
PDF417	Portable Data File 417
PLANET	PostaL Alpha Numeric Encoding Technique (United States Postal Service)
POSTNET	POSTal Numeric Encoding Technique (United States Postal Service)
QR Code	Quick Response Code
RM4SCC	Royal Mail 4 State Customer Code
UCC	Uniform Code Council
UPC	Universal Product Code (United States)
UPC/CGPC	Universal Product Code (United States) and the Canadian Grocery Product Code
USS	Uniform Symbol Specification

Byte 13 MOD

The modifier field gives additional processing information about the bar code symbol to be generated. For example, it indicates whether a check-digit is to be generated for the bar code symbol. The check digit algorithm and placement are defined in Appendix A, "Bar Code Symbology Specification References," on page 109. Exception condition EC-0B00 exists if the bar code modifier is invalid or unsupported for the bar code type specified.

Table 3 defines the BCD1 bar code modifier codes that must be supported for each bar code type specified.

Table 3. Modifier Values by Bar Code Type

Bar Code Type (byte 12)	Modifier Value (byte 13)	In BCD1 Subset?
X'01' – Code 39 (3-of-9 Code), AIM USS-39	X'01' and X'02'	Yes
X'02' – MSI (modified Plessey code)	X'01' through X'09'	Yes
X'03' – UPC/CGPC Version A	X'00'	Yes
X'05' – UPC/CGPC Version E	X'00'	Yes
X'06' – UPC - Two-digit Supplemental	X'00'	Yes
X'06' – UPC - Two-digit Supplemental	X'01' and X'02'	No
X'07' – UPC - Five-digit Supplemental	X'00'	Yes
X'07' – UPC - Five-digit Supplemental	X'01' and X'02'	No
X'08' – EAN 8 (includes JAN-short)	X'00'	Yes
X'09' – EAN 13 (includes JAN-standard)	X'00'	Yes
X'0A' – Industrial 2-of-5	X'01' and X'02'	No
X'0B' – Matrix 2-of-5	X'01' and X'02'	No
X'0C' – Interleaved 2-of-5, AIM USS-I 2/5	X'01' and X'02'	Yes
X'0D' – Codabar, 2-of-7, AIM USS-Codabar	X'01' and X'02'	No
X'11' – Code 128, AIM USS-128	X'02' and X'03'	No
X'16' – EAN Two-digit Supplemental	X'00'	Yes
X'16' – EAN Two-digit Supplemental	X'01'	No
X'17' – EAN Five-digit Supplemental	X'00'	Yes
X'17' – EAN Five-digit Supplemental	X'01'	No
X'18' – POSTNET	X'00' through X'04'	No
X'1A' – RM4SCC	X'00' and X'01'	No
X'1B' – Japan Postal Bar Code	X'00' and X'01'	No
X'1C' – Data Matrix (2D bar code)	X'00'	No
X'1D' – MaxiCode (2D bar code)	X'00'	No
X'1E' – PDF417 (2D bar code)	X'00' and X'01'	No
X'1F' – Australia Post Bar Code	X'01' through X'08'	No
X'20' — QR Code	X'02'	No
X'21' — Code 93	X'00'	No

The assigned values of this field, by bar code type, are as follows:

Code 39 (3-of-9 Code), AIM USS-39

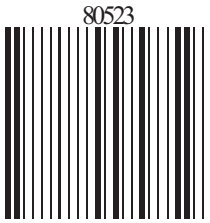
Code 39 (3-of-9 Code)
(encoding 39OR93 with check character
yielding a 2.32 inch wide symbol)

X'01' Present the bar code without a generated check digit.

X'02' Generate a check digit and present it with the bar code.

Note: The Code 39 character set contains 43 characters including numbers, upper-case alphabetics, and some special characters. The Code 39 Specification also provides a method of encoding all 128 ASCII characters by using 2 bar code characters for those ASCII characters that are not in the standard Code 39 character set. This is sometimes referred to as “Extended Code 39” and is supported by all BCOCA receivers. In this case, the 2 bar code characters used to specify the “extended character” will be shown in the Human-Readable Interpretation and the bar code scanner will interpret the 2-character combination bar/space pattern appropriately.

MSI (modified Plessey code)



MSI - no check digit
(encoding 80523)

X'01' Present the bar code without check digits generated by the printer. Specify 3 to 15 digits of input data.

X'02' Present the bar code with a generated IBM modulo-10 check digit. This check digit will be the second check digit; the first check digit is the last byte of the BSA data. Specify 2 to 14 digits of input data.

X'03' Present the bar code with two check digits. Both check digits are generated using the IBM modulo-10 algorithm. Specify 1 to 13 digits of input data.

X'04' Present the bar code with two check digits. The first check digit is generated using the NCR modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals the remainder; exception condition EC-0E00 exists if the first check-digit calculation results in a value of 10. Specify 1 to 13 digits of input data.

X'05' Present the bar code with two check digits. The first check digit is generated using the IBM modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals the remainder; exception condition EC-0E00 exists if the first check-digit calculation results in a value of 10. Specify 1 to 13 digits of input data.

X'06' Present the bar code with two check digits. The first check digit is generated using the NCR modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals 11 minus the remainder; a first check digit value of 10 is assigned the value zero. Specify 1 to 13 digits of input data.

X'07' Present the bar code with two check digits. The first check digit is generated using the IBM modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals 11 minus the remainder; a first check digit value of 10 is assigned the value zero. Specify 1 to 13 digits of input data.

X'08' Present the bar code with two check digits. The first check digit is generated using the NCR modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals 11 minus the remainder; exception condition EC-0E00 exists if the

first check-digit calculation results in a value of 10. Specify 1 to 13 digits of input data.

X'09' Present the bar code with two check digits. The first check digit is generated using the IBM modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals 11 minus the remainder; exception condition EC-0E00 exists if the first check-digit calculation results in a value of 10. Specify 1 to 13 digits of input data.

UPC/CGPC—Version A

UPC Version A
(encoding 01234567890)

X'00' Present the standard UPC-A bar code with a generated check digit. The data to be encoded consists of eleven digits. The first digit is the number-system digit; the next ten digits are the article number.

Specify 11 digits of input data. The first digit is the number system character; the remaining digits are information characters.

UPC/CGPC—Version E

UPC Version E
(encoding 078349)

X'00' Present a UPC-E bar code symbol. Of the 10 input digits, six digits are encoded. The check digit is generated using all 10 input data digits. The check digit is not encoded; it is only used to assign odd or even parity to the six encoded digits.

Specify 10 digits of input data. Version E suppresses some zeros that can occur in the information characters to produce a shorter symbol. All 10 digits are information characters; the number system character should not be specified (it is assumed to be 0).

UPC—Two-Digit Supplemental

UPC A + Two-digit Supplemental
(encoding 00633895260, supplemental = 24)

X'00' Present a UPC two-digit supplemental bar code symbol. This option assumes that the base UPC Version A or E symbol is presented as a separate bar code object. The bar and space patterns used for the two supplemental digits are left-odd or left-even parity, with the parity determined by the digit combination.

Specify 2 digits of input data.

X'01' The two-digit UPC supplemental bar code symbol is preceded by a UPC Version A, Number System 0, bar code symbol. The bar code object contains both the UPC Version A symbol and the two-digit supplemental symbol. The input data consists of the number system digit, the ten-digit article number, and the two supplement digits, in that order. A check digit is generated for the UPC Version A symbol. The two-digit supplemental bar code is presented after the UPC Version A symbol using left-hand odd and even parity as determined by the two supplemental digits.

Specify 13 digits of input data.

X'02' The two-digit UPC supplemental bar code symbol is preceded by a UPC Version E symbol. The bar code object contains both the UPC Version E symbol and the two-digit supplemental symbol. The input data consists of the ten-digit article number and the two supplemental digits. The bar code object processor generates the six-digit UPC Version E symbol and a check digit. The check digit is used to determine the parity pattern of the six-digit Version E symbol. The two-digit supplemental bar code symbol is presented after the Version E symbol using left-hand odd and even parity as determined by the two digits.

Specify 12 digits of input data.

UPC—Five-Digit Supplemental

UPC A + Five-digit Supplemental
(encoding 09827721123, supplemental = 21826)

X'00' Present the UPC five-digit supplemental bar code symbol. This option assumes that the base UPC Version A or E symbol is presented as a separate bar code object. A check digit is generated from the five supplemental digits and is used to assign the left-odd and left-even parity of the five-digit supplemental bar code. The supplemental check digit is not encoded or interpreted.

Specify 5 digits of input data.

X'01' The five-digit UPC supplemental bar code symbol is preceded by a UPC Version A, Number System 0, bar code symbol. The bar code object contains both the UPC Version A symbol and the five-digit supplemental symbol. The input data consists of the number system digit, the ten-digit article number, and the five supplement digits, in that order. A check digit is generated for the UPC Version A symbol. A second check digit is generated from the five supplement digits. It is used to assign the left-hand odd and even parity of the five-digit supplemental bar code symbol. The supplement check digit is not encoded or interpreted.

Specify 16 digits of input data.

X'02' The five-digit UPC supplemental bar code symbol is preceded by a UPC Version E symbol. The bar code object contains both the UPC Version E symbol and the five-digit supplemental symbol. The input data consists of the ten-digit article number and the five-digit supplemental data. The bar code object processor generates the six-digit UPC Version E symbol and check digit. The check digit is used to determine the parity pattern of the Version E symbol. The five-digit supplemental bar code symbol is presented after the Version E symbol. A second check digit is calculated for the five-digit supplemental data and is used to assign the left-hand odd and even parity. The supplement check digit is not encoded or interpreted.

Specify 15 digits of input data.

EAN-8 (includes JAN-short)

EAN 8
(encoding 2468123)

X'00' Present an EAN-8 bar code symbol. The input data consists of seven digits: two flag digits and five article number digits. All seven digits are encoded along with a generated check digit.

EAN-13 (includes JAN-standard)

EAN 13
(encoding 041234567890)

X'00' Present an EAN-13 bar code symbol. The input data consists of twelve digits: two flag digits and ten article number digits, in that order. The first flag digit is not encoded. The second flag digit, the article number digits, and generated check digit are encoded. The first flag digit is presented in HRI form at the bottom of the left *quiet zone*. The first flag digit governs the A and B number-set pattern of the bar and space coding of the six digits to the left of the symbol center pattern.

Industrial 2-of-5

Industrial 2-of-5
(encoding 54321068)

X'01' Present the bar code without a generated check digit.

X'02' Generate a check digit and present it with the bar code.

Matrix 2-of-5

Matrix 2-of-5
(encoding 54321068)

X'01' Present the bar code symbol without a generated check digit.

X'02' Generate a check digit and present it with the bar code.

Interleaved 2-of-5, AIM USS-I 2/5

Interleaved 2-of-5
(encoding 54321068)

The Interleaved 2-of-5 symbology requires an even number of digits, and the printer will add a leading zero if necessary to meet this requirement.

X'01' Present the bar code symbol without a check digit.

X'02' Generate a check digit and present it with the bar code.

Codabar, 2-of-7, AIM USS-Codabar

Codabar
(encoding A34698735B)

X'01' Present the bar code without a generated check digit. The input data consists of a start character, digits to be encoded, and a stop character, in that order. Start and stop characters can be A, B, C, or D, and can only be used at the beginning and end of the symbol.

X'02' Generate a check digit and present it with the bar code. The input data consists of a start character, digits to be encoded, and a stop character, in that order. Start and stop characters can be A, B, C, or D, and can only be used at the beginning and end of the symbol.

Code 128, AIM USS-128

The 1986 symbology definition for Code 128 defined an algorithm for generating a start character and then changed that algorithm in 1993 to accommodate the UCC/EAN 128 variation of this bar code. Many BCOCA printers have implemented the 1986 version (using modifier X'02'), some BCOCA printers have changed to use the 1993 algorithm (with modifier X'02'), and some BCOCA printers support both algorithms. When producing UCC/EAN 128 bar codes for printers that explicitly support UCC/EAN 128, modifier X'03' should be specified. For printers that do not explicitly support UCC/EAN 128, specifying modifier X'02' might produce a valid UCC/EAN 128 bar code (see notes in the modifier descriptions).

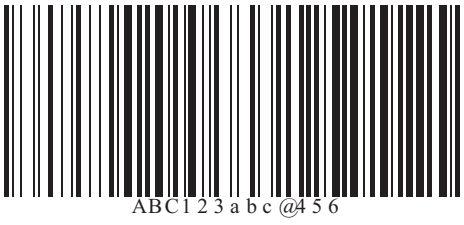
The data for UCC/EAN 128 bar codes is in the form:

```
"FN1, ai, data, m, FN1, ai, data, m, FN1, ..., ai, data, m"
```

where “FN1” is the FN1 function character (X'8F'), “ai” is an application identifier, “data” is defined for each registered application identifier, and “m” is a modulo 10 check digit (calculated using the same check digit algorithm as is used for UPC version A bar codes); note that not all application identifiers require a modulo 10 check digit (m). Also, note that all except the first “FN1” are field separator characters that only appear when the preceding ai data is of variable length. Refer to *UCC/EAN-128 APPLICATION IDENTIFIER STANDARD* from the Uniform Code Council, Inc. for a description of application identifiers and the use of “FN1”. When building the bar code symbol, the printer will:

1. produce a start character based on the 1993 algorithm
2. bar encode the data including all of the “FN1”, “ai”, “data”, and “m” check digit
3. produce a modulo 103 check digit
4. produce a stop character.

X'02' Code 128 symbol using original start-character algorithm



Code 128
(encoding ABC123abc@456)

Generate a Code 128 symbol using subset A, B, or C as appropriate to produce the shortest possible bar code from the given data, using the start-character algorithm that was published in the original (1986) edition of the Code 128 Symbology Specification. The Code 128 code page (CPGID = 1303, GCSGID = 1454) is used to interpret the bar code symbol data. Generate a check digit and present it with the bar code.

Notes:

1. Some IPDS printers use the modifier X'03' start-character algorithm even when modifier X'02' is specified; this produces a valid UCC/EAN 128 symbol when valid UCC/EAN 128 data is provided. However, in general, modifier X'02' should not be used to produce UCC/EAN 128 symbols since this value causes other IPDS printers to use the original Code 128 start-symbol algorithm which will generate a Start (Code B) instead of the Start (Code C) that UCC/EAN 128 requires. Some bar code scanners can handle either start character for a UCC/EAN 128 symbol, but others require the Start (Code C) character.
2. Printers that use the UCC/EAN 128 start-character algorithm when modifier X'02' is specified include: 4312, 4317, 4324, Infoprint 20, Infoprint 21, Infoprint 32, Infoprint 40, Infoprint 45, Infoprint 70, Infoprint 2070, Infoprint 2085, and Infoprint 2105. Other IPDS printers provided by IBM use the original start-character algorithm when modifier X'02' is specified.

X'03' Code 128 symbol using the start-character algorithm that supports UCC/EAN 128

UCC/EAN 128
(encoding FNC1, 00, 0, 1111111, 222222222, 9)

Generate a Code 128 symbol using subset A, B, or C as appropriate to produce the shortest possible bar code from the given data, using the version of the start-character algorithm that was modified for producing UCC/EAN 128 symbols. If the first data character is FN1 (as is required for a UCC/EAN 128 symbol) and is followed by valid UCC/EAN 128 data, the printer will generate a Start (Code C) character. The Code 128 code page (CPGID = 1303, GCSGID = 1454) is used to interpret the bar code symbol data. Generate a check digit and present it with the bar code.

Note: UCC/EAN 128 is a variation of Code 128 that begins with a FN1 character, followed by an Application Identifier and the data to be bar encoded. All of these characters (including the FN1 character) must be supplied within the Bar Code Symbol Data (BSA). UCC/EAN 128 also requires that the symbol begin in subset C.

EAN Two-Digit Supplemental

EAN + 2 Digit Supplemental
(encoding 041234567890, supplemental = 99)

X'00' Present the EAN two-digit supplemental bar code symbol. This option assumes that the base EAN-13 symbol is presented as a separate bar code object. The value of the two digit supplemental data determines their bar and space patterns chosen from number sets A and B.

Specify 2 digits of input data.

X'01' The two-digit supplemental bar code symbol is preceded by a normal EAN-13 bar code symbol. The bar code object contains both the EAN-13 symbol and the two-digit supplemental symbol. The two-digit supplemental bar code is presented after the EAN-13 symbol using left hand odd and even parity as determined by the two supplemental digits chosen from number sets A and B.

Specify 14 digits of input data.

Note: Used for both books and paperbacks.

EAN Five-Digit Supplemental

EAN + 5 Digit Supplemental
(encoding 041234567890, supplemental = 54321)

X'00' Present the EAN five-digit supplemental bar code. This option assumes that the base EAN-13 symbol is presented as a separate bar code object. A check digit is calculated from the five supplemental digits. The check digit is also used to assign the bar and space patterns from number sets A and B for the five supplemental digits. The check digit is not encoded or interpreted.

Specify 5 digits of input data.

X'01' The five-digit supplemental bar code symbol is preceded by a normal EAN-13 bar code symbol. The bar code object contains both the EAN-13 symbol and the five-digit supplemental symbol. A check digit is generated from the five-digit supplemental data. The check digit is used to assign the bar and space patterns from number sets A and B. The check digit is not encoded or interpreted.

Specify 17 digits of input data.

Note: Used for books and paperbacks.

POSTNET

US POSTNET
Zip+4
(encoding 12345+6789)

PLANET Code
(encoding 00123456789)

For all POSTNET modifiers that follow, the BSA HRI flag field and the BSD module width, element height, height multiplier, and wide-to-narrow ratio fields are not applicable to the POSTNET bar code symbology. These fields are ignored because the POSTNET symbology defines specific values for these parameters.

X'00' Present a POSTNET ZIP Code bar code symbol. The ZIP Code to be encoded is defined as a five-digit, numeric (0–9), data variable to the BSA data structure. The POSTNET ZIP Code bar code consists of a leading frame bar, the encoded ZIP Code data, a correction digit, and a trailing frame bar.

X'01' Present a POSTNET ZIP+4 bar code symbol. The ZIP+4 code to be encoded is defined as a nine-digit, numeric (0–9), data variable to the BSA data structure. The POSTNET ZIP+4 bar code consists of a leading frame bar, the encoded ZIP+4 data, a correction digit, and a trailing frame bar.

X'02' Present a POSTNET Advanced Bar Code (ABC) bar code symbol. The ABC code to be encoded is defined as an eleven-digit, numeric (0–9), data variable to the BSA data structure. The POSTNET ABC bar code consists of a leading frame bar, the encoded ABC data, a correction digit, and a trailing frame bar.

Note: An 11-digit POSTNET bar code is called a *Delivery Point bar code*.

X'03' Present a POSTNET variable-length bar code symbol. The data to be encoded is defined as an n-digit, numeric (0–9), data variable to the BSA data structure. The bar code symbol is generated without length checking; the symbol is not guaranteed to be scannable or interpretable. The POSTNET variable-length bar code consists of a leading frame bar, the encoded data, a correction digit, and a trailing frame bar.

X'04' Present a PLANET Code symbol. The PLANET Code is a reverse topology variation of POSTNET that encodes 11 digits of data; the first 2 digits represent a service code (such as, 21 = Origin Confirm and 22 = Destination Confirm) and the next 9 digits identify the mailpiece. A 12th digit is generated by the printer as a check digit. The PLANET Code symbol consists of a leading frame bar, the encoded data, a check digit, and a trailing frame bar.

Royal Mail (RM4SCC)

Royal Mail (RM4SCC)
UK and Singapore version
(encoding SN34RD1A)

Royal Mail (RM4SCC)
Dutch KIX version
(encoding SN34RD1A)

A 4-state customer code defined by the Royal Mail Postal service of England for use in bar coding postal code information. This symbology is also called the *Royal Mail bar code* or the *4-State customer code*. The symbology (as defined for modifier X'00') is used in the United Kingdom and in Singapore. A variation called KIX (KlantenIndeX = customer index, as defined for modifier X'01') is used in the Netherlands.

X'00' Present a RM4SCC bar code symbol with a generated start bar, checksum character, and stop bar. The start and stop bars identify the beginning and end of the bar code symbol and also the orientation of the symbol.

X'01' Dutch KIX variation – Present a RM4SCC bar code symbol with no start bar, no checksum character, and no stop bar.

Japan Postal Bar Code

Japan Postal Bar Code
Modifier X'00'
(encoding 15400233-16-4)

A bar code symbology defined by the Japanese Postal Service for use in bar coding postal code information.

X'00' Present a Japan Postal Bar Code symbol with a generated start character, checksum character, and stop character.

The generated bar code symbol will consist of a start code, a 7-digit new postal code, a 13-digit address indication number, a check digit, and a stop code. The variable data to be encoded (BSA bytes 5–n) will be used as follows:

1. The first few digits is the new postal code in either the form nnn-nnnn or the form nnnnnnn; the hyphen, if present, is ignored and the other 7 digits must be numeric. These 7 digits will be placed in the new postal code field of the bar code symbol.
2. If the next character is a hyphen, it is ignored and is not used in generating the bar code symbol.
3. The remainder of the BSA data is the address indication number which can contain numbers, hyphens, and alphabetic characters (A–Z). Each number and each hyphen represents one digit in the bar code symbol; each alphabetic character is represented by a combination of a control code (CC1, CC2, or CC3) and a numerical code and shall be handled as two digits in the bar code symbol. 13 digits of this address indication number data will be placed in the address indication number field of the bar code symbol.
   - If less than 13 additional digits are present, the shortage shall be filled in with the bar code corresponding to control code CC4 up to the 13th digit.
   - If more than 13 additional digits are present, the first 13 digits will be used and the remainder ignored with no exception condition reported. However, if the 13th digit is the control code for an alphabetic (A–Z) character, only the control code is included and the numeric part is omitted.

X'01' Present a Japan Postal Bar Code symbol directly from the bar code data.

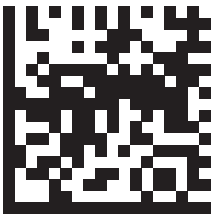
Each valid character in the BSA data field is converted into a bar/space pattern with no validity or length checking. The printer will not generate start, stop, and check digits.

To produce a valid bar code symbol, the bar code data must contain a start code, a 7-digit new postal code, a

13-digit address indication number, a valid check digit, and a stop code. The new postal code must consist of 7 numeric digits. The address indication number must consist of 13 characters which can be numeric, hypen, or control characters (CC1 through CC8). The following table lists the valid code points for modifier X'01'.

Character	Code Point		Character	Code Point
start	X'4C'		0	X'F0'
stop	X'6E'		1	X'F1'
hyphen	X'60'		2	X'F2'
CC1	X'5A'		3	X'F3'
CC2	X'7F'		4	X'F4'
CC3	X'7B'		5	X'F5'
CC4	X'E0'		6	X'F6'
CC5	X'6C'		7	X'F7'
CC6	X'50'		8	X'F8'
CC7	X'7D'		9	X'F9'
CC8	X'4D'			

Data Matrix



Data Matrix 2D Symbol
(encoding A1B2C3D4E5F6G7H8I9J0K1L2)

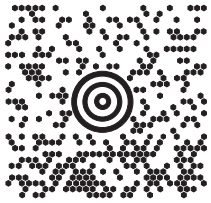
A two-dimensional matrix bar code symbology defined as an AIM International Symbol Specification.

X'00' Present a Data Matrix Bar Code symbol using Error Checking and Correcting (ECC) algorithm 200.

The bar code data is assumed to start with the default character encodation (ECI 000003 = ISO 8859-1). This is an international Latin 1 code page that is equivalent to the IBM ASCII code page 819. To change to a different character encodation within the data, the ECI protocol as defined in the *AIM International Symbology Specification - Data Matrix*, must be used. This means that whenever a byte value of X'5C' (an escape code) is encountered in the bar code data, the next six characters must be decimal digits (byte values X'30' to X'39') or the next character must be another X'5C'. When the X'5C' character is followed by six decimal digits, the six decimal digits are interpreted as the ECI number which changes the interpretation of the characters that follow the decimal digits. When the X'5C' character is followed by another X'5C' character, this is interpreted as one X'5C' character (which is a backslash in the default character encodation); alternatively, the escape-sequence handling flag (see page 71) can be used to treat X'5C' as a normal character.

Since the default character encodation for this bar code is ASCII, the EBCDIC-to-ASCII translation flag (see page 71) can be used when all of the data for the bar code is EBCDIC. If the bar code data contains more than one character encodation or if the data needs to be encoded within the bar code symbol in a form other than the default character encodation (such as, in EBCDIC), the bar code data should begin in the default encodation, the EBCDIC-to-ASCII translation flag should be set to B'0', and the ECI protocol should be used to switch into the other encodation.

MaxiCode



MaxiCode 2D Symbol

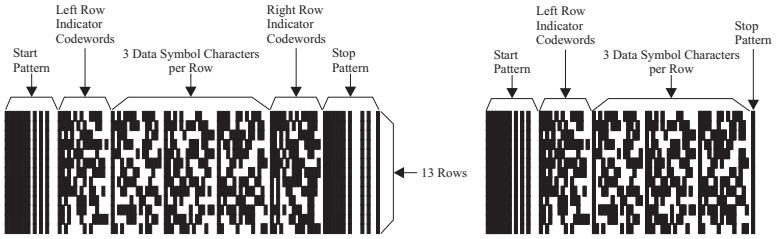
A two-dimensional matrix bar code symbology as defined in the *AIM International Symbology Specification – MaxiCode*.

X'00' Present a MaxiCode bar code symbol.

The bar code data is assumed to start with the default character encodation (ECI 000003 = ISO 8859-1). This is an international Latin 1 code page that is equivalent to the IBM ASCII code page 819. To change to a different character encodation within the data, the ECI protocol as defined in section 4.15.2 of the *AIM International Symbology Specification - MaxiCode*, must be used. This means that whenever a byte value of X'5C' (an escape code) is encountered in the bar code data, the next six characters must be decimal digits (byte values X'30' to X'39') or the next character must be another X'5C'. When the X'5C' character is followed by six decimal digits, the six decimal digits are interpreted as the ECI number which changes the interpretation of the characters that follow the decimal digits. When the X'5C' character is followed by another X'5C' character, this is interpreted as one X'5C' character (which is a backslash in the default character encodation); alternatively, the escape-sequence handling flag (see page 75) can be used to treat X'5C' as a normal character. The X'5C' character is allowed anywhere in the bar code data except for Modes 2 and 3 where it is not allowed in the Primary Message portion of the data.

Since the default character encodation for this bar code is ASCII, the EBCDIC-to-ASCII translation flag (see page 75) can be used when all of the data for the bar code is EBCDIC. If the bar code data contains more than one character encodation or if the data needs to be encoded within the bar code symbol in a form other than the default character encodation (such as, in EBCDIC), the bar code data should begin in the default encodation, the EBCDIC-to-ASCII translation flag should be set to B'0', and the ECI protocol should be used to switch into the other encodation.

PDF417



PDF417 Truncated PDF417

A two-dimensional stacked bar code symbology as defined in the *AIM Uniform Symbology Specification – PDF417*.

X'00' Present a full PDF417 bar code symbol.

X'01' Present a truncated PDF417 bar code symbol, for use in a relatively clean environment in which damage to the symbol is unlikely. This version omits the right row indicator and simplifies the stop pattern into a single module width bar.

The bar code data is assumed to start with the default character encodation (GLI 0) as defined in Table 5 of the Uniform Symbology Specification PDF417. To change to another character encodation, the GLI (Global Label Identifier) protocol, as defined in the Uniform Symbology Specification PDF417, must be used. This means that whenever a byte value of X'5C' (an escape code) is encountered in the bar code data, the next three characters must be decimal digits (byte values X'30' to X'39') or the next character must be another X'5C' character. When the X'5C' character is followed by three decimal digits, this is called an escape sequence. When the X'5C' character is followed by another X'5C' character, this is interpreted as one X'5C' character (which is a backslash in the default character encodation); alternatively, the escape-sequence handling flag (see page 83) can be used to treat X'5C' as a normal character.

To identify a new GLI, there must be two or three escape sequences in a row. The first escape sequence must be “\925”, “\926”, or “\927” (as defined by GLI 0). If the first escape sequence is “\925” or “\927”, there must be one other escape sequence following containing a value from “\000” to “\899”. If the first escape sequence is “\926”, there must be two more escape sequences following with each escape sequence containing a value from “\000” to “\899”. For example, to switch to GLI 1 (ISO 8859-1 which is equivalent to IBM ASCII code page 819), the bar code data would contain the character sequence “\927\001”. The “\927” escape sequence is used for GLI values from 0 to 899. The “\926” escape sequence is used for GLI values from 900 to 810,899. The “\925” escape sequence is used for GLI values from 810,900 to 811,799. For more information about how these values are calculated refer to section 2.2.6 of the Uniform Symbology Specification PDF417.

In addition to transmitting GLI numbers, the escape sequence is used to transmit other codewords for additional purposes. The

special codewords are given in Table 8 in Section 2.7 of the Uniform Symbology Specification PDF417. The special codewords “\903” to “\912” and “\914” to “\920” are reserved for future use. The BCOCA receiver will accept these special escape sequences and add them to the bar code symbol, resuming with normal encoding with the character following that escape sequence.

The special codeword “\921” instructs the bar code reader to interpret the data contained within the symbol for reader initialization or programming. This escape sequence is only allowed at the beginning of the bar code data.

The special codewords “\922”, “\923”, and “\928” are used for coding a Macro PDF417 Control Block as defined in section G.2 of the Uniform Symbology Specification PDF417. These codewords must not be used within the BCOCA data; instead a Macro PDF417 Control Block can be specified in the special-function parameters. Exception condition EC-2100 exists if one of these escape sequences is found in the bar code data.

Since the default character encodation for this bar code is GLI 0 (an ASCII code page that is similar to IBM code page 437), the EBCDIC-to-ASCII translation flag (see page 81) can be used when all of the data for the bar code is EBCDIC. If the bar code data contains more than one character encodation, or if the data needs to be encoded within the bar code symbol in a form other than the default character encodation (such as, in EBCDIC), the bar code data should begin in the default encodation, the EBCDIC-to-ASCII translation flag should be set to B'0', and the GLI protocol should be used to switch into the other encodation.

Australia Post Bar Code

Australia Post Bar Code
Customer Barcode 2 using Table C
(encoding 56439111ABA 9)

A bar code symbology defined by Australia Post for use in Australian postal systems. There are several formats of this bar code, which are identified by the modifier byte as follows:

Modifier	Type of Bar Code	Valid Bar Code Data
X'01'	Standard Customer Barcode (format code = 11)	An 8 digit number representing the Sorting Code
X'02'	Customer Barcode 2 using Table N (format code = 59)	An 8 digit number representing the Sorting Code followed by up to 8 numeric digits representing the Customer Information
X'03'	Customer Barcode 2 using Table C (format code = 59)	An 8 digit number representing the Sorting Code followed by up to 5 characters (A–Z, a–z, 0–9, space, #) representing the Customer Information
X'04'	Customer Barcode 2 using proprietary encoding (format code = 59)	An 8 digit number representing the Sorting Code followed by up to 16 numeric digits (0–3) representing the Customer Information. Each of the 16 digits specify one of the 4 types of bar.
X'05'	Customer Barcode 3 using Table N (format code = 62)	An 8 digit number representing the Sorting Code followed by up to 15 numeric digits representing the Customer Information
X'06'	Customer Barcode 3 using Table C (format code = 62)	An 8 digit number representing the Sorting Code followed by up to 10 characters (A–Z, a–z, 0–9, space, #) representing the Customer Information
X'07'	Customer Barcode 3 using proprietary encoding (format code = 62)	An 8 digit number representing the Sorting Code followed by up to 31 numeric digits (0–3) representing the Customer Information. Each of the 31 digits specify one of the 4 types of bar.
X'08'	Reply Paid Barcode (format code = 45)	An 8 digit number representing the Sorting Code

The proprietary encoding allows the customer to specify the types of bars to be printed directly by using 0 for a full bar, 1 for an ascending bar, 2 for a descending bar and 3 for a timing bar. If the customer does not specify enough Customer Information to fill the field, the printer uses a filler bar to extend pad the field out to the correct number of bars.

The printer will encode the data using the proper tables, generate the start and stop bars, generate any needed filler bars, and generate the Reed Solomon ECC bars.

Human-readable interpretation (HRI) can be selected with this bar code type. The format control code, Delivery Point Identifier, and customer information field (if any) appears in the HRI, but the ECC does not.

QR Code

QR Code 2D Symbol

A two-dimensional matrix bar code symbology defined as an AIM International Technical Standard.

X'02' Present a Model 2 QR Code Bar Code symbol as defined in *AIM International Symbology Specification — QR Code*.

The bar code data is assumed to start with the default character encodation (ECI 000020). This is a single–byte code page representing the JIS8 and Shift JIS character sets; it is equivalent to the IBM ASCII code page 897. To change to a different character encodation within the data, the ECI protocol as defined in the *AIM International "Extended Channel Interpretation (ECI) Assignments"*, must be used.

Since the default character encodation for this bar code is ASCII, the EBCDIC-to-ASCII translation flag (see page 88) can be used when all of the data for the bar code is single–byte EBCDIC. If the bar code data contains more than one character encodation or if the data needs to be encoded within the bar code symbol in a form other than the default character encodation (such as, in EBCDIC), the bar code data should begin in the default encodation, the EBCDIC-to-ASCII translation flag should be set to B'0', and the ECI protocol should be used to switch into the other encodation.

There must be a quiet zone around the symbol that is at least 4 modules wide on each of the four sides of the symbol.

Code 93

Code 93
(encoding 39OR93
yielding a 1.82 inch wide symbol)

A linear bar code symbology similar to Code 39, but more compact than Code 39. Code 93 bar code symbols are made up of a series of characters each of which is represented by 9 modules arranged into 3 bars with their adjacent spaces. The bars and spaces vary between 1 module wide and 4 modules wide.

X'00' Present a Code 93 bar code symbol as defined in *AIM Uniform Symbology Specification — Code 93*.

The Code 93 character set contains 47 characters including numeric digits, upper-case alphabetics, four shift characters (a,b,c,d), and seven special characters. The Code 93 Specification also provides a method of encoding all 128 ASCII characters by using 2 bar code characters for those ASCII characters that are not in the standard Code 93 character set. This is sometimes referred to as "Extended Code 93". In this case, the 2 bar code characters used to specify the "extended character" will be shown in the Human-Readable Interpretation (as a ■ followed by the second character) and the bar code scanner will interpret the two-character combination bar/space pattern appropriately.

The Human-Readable Interpretation of the Start and Stop characters is represented as an open box (□) and the shift characters (a,b,c,d) are represented as a filled box (■).

There must be a quiet zone preceding and following the symbol that is at least 10 modules wide.

Byte 14 LID

Specifies the local ID of a font to be used when HRI is requested. A value of X'FF' indicates that a presentation device selected font is to be used. Since most BCOCA receivers provide resident font resources for use with the supported bar code symbologies, specifying a local ID of X'FF' is recommended.

Some bar code symbology specifications do not specify a type style for HRI information. However, the UPC and EAN symbologies specify OCR-B for HRI; refer to Table 7 on page 93. The location of the HRI is specified and varies depending on the symbology selected.

For bar code types that do not allow HRI information, for example, Data Matrix, Japan Postal Bar Code, MaxiCode, PDF417, POSTNET, QR Code, and RM4SCC, this field is ignored.

For those symbologies that require a specific type style or code page for HRI, exception condition EC-0400 exists if the printer cannot determine the type style or code page of the specified font.

Note: Specifying LID = X'FF' is the easiest way to guarantee that a proper font is selected. If another LID is specified, the font must be appropriate for the specified symbology; using a printer-resident font is recommended in this case.

Exception condition EC-0400 exists if a local ID is unsupported or the font is not available. If the requested font is not available, a substitution can be made that preserves as many characteristics as possible of the originally requested font; the code page selected must be a superset of the requested code page. Otherwise, terminate bar code object processing and continue with the next object.

Some bar code symbologies specify a set of type styles to be used for HRI data. Font substitution for HRI data must follow the bar code symbology specification being used.

Bytes 15–16 COLOR

Specifies the color in which the bars of the bar code symbol and the HRI is to be presented. Valid values for specifying color include the OCA standard color values (X'0000'–X'0010' and X'FF00'–X'FF08') shown in Table 4 and the special value X'FFFF' which selects the device default color. Exception condition EC-0500 exists if the color specified is invalid or unsupported. If the color is unsupported, the presentation device default color is used. Some devices simulate an unsupported color without reporting an exception condition.

Table 4. Standard OCA Color-Value Table

Value	Color	Red (R)	Green (G)	Blue (B)
X'0000' or X'FF00'	Device default			
X'0001' or X'FF01'	Blue	0	0	255
X'0002' or X'FF02'	Red	255	0	0
X'0003' or X'FF03'	Pink/magenta	255	0	255
X'0004' or X'FF04'	Green	0	255	0
X'0005' or X'FF05'	Turquoise/cyan	0	255	255
X'0006' or X'FF06'	Yellow	255	255	0
X'0007'	White; see note 1	255	255	255
X'0008'	Black	0	0	0
X'0009'	Dark blue	0	0	170
X'000A'	Orange	255	128	0
X'000B'	Purple	170	0	170
X'000C'	Dark green	0	146	0
X'000D'	Dark turquoise	0	146	170
X'000E'	Mustard	196	160	32
X'000F'	Gray	131	131	131
X'0010'	Brown	144	48	0
X'FF07'	Device default			
X'FF08'	Color of medium; also known as reset color			
Note: The table specifies the RGB values for each named color; the actual printed color is device dependent.				

Notes:

1. 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 when white media is used.

2. Some symbologies, such as Data Matrix, allow the bar code symbol to be presented in a reverse video manner (light modules on a dark background). To achieve this effect, color the bar code object area with a dark color and specify color of medium (X'FF08') for the symbol color. In a MO:DCA environment, the bar code object area can be colored using a Color Specification triplet in the Object Area Descriptor. In an IPDS environment, the bar code object area can be colored using a Color Specification triplet in the Bar Code Output Control.

Byte 17 MODULE WIDTH

This parameter specifies the width in mils (thousandths of an inch) of the smallest defined bar code element (bar, space, or 2D module). Some bar code symbologies refer to this value as the unit or X-dimension width. The widths of all symbol elements are normally expressed as multiples (not necessarily integer multiples) of the module width. A value of X'FF' indicates the default module width of the presentation device is to be used. Exception condition EC-0600 exists if the module width specified is invalid or unsupported. For this condition, the bar code object processor uses the closest smaller width. If the smaller value is less than the smallest supported width or zero, the bar code object processor uses the smallest supported value.

For bar code types that explicitly specify the module width, for example, Australia Post Bar Code, MaxiCode, POSTNET, and RM4SCC, this field is ignored.

Presentation devices must map the module width specification in mils to an integer number of device pels. This mapping yields an approximation of the user request and causes the total width of a bar code symbol to be different at different device resolutions.

The following equations can be used to convert between L-units, mils, and millimeters, where:
X is the symbol for multiplication
/ is the symbol for division

1. Inches **X** (units per unit base) = L-units, also
   L-units/(units per unit base) = inches
   For example, when units per unit base is 1440ths:
   Inches **X** 1440 = L-units
2. Inches **X** 1000 = mils, also mils/1000 = inches
3. Inches **X** 25.4 = mm, also mm/25.4 = inches

From (1), (2), and (3) above, using units per unit base of 1440:
mils **X** 1.44 = L-units
mm **X** 1440/25.4 = L-units

Since the modules for a Data Matrix symbol and a QR Code symbol are defined to be square, the module width parameter specifies both dimensions, and the element height and height multiplier parameters are not used for these symbologies.

Bytes 18–19 ELEMENT HEIGHT

Specifies the height in L-units along the Y_{bc} axis of the bar code symbol bar elements. The measurement unit base is specified in BSD bytes 0–5. The element height and height multiplier values are used to specify the total bar height presented. The height of the HRI is not included in this total height for many bar code symbologies; however, for the following symbologies, the total symbol height includes both bar patterns as well as the HRI:

- UPC/CGPC Version A, modifier X'00'
- UPC/CGPC Version E, modifier X'00'
- UPC Two-digit supplemental, modifiers X'01' and X'02' (the total height applies to the main symbol; the height of the supplement is calculated from the the main-symbol height)
- UPC Five-digit supplemental, modifiers X'01' and X'02' (the total height applies to the main symbol; the height of the supplement is calculated from the the main-symbol height)
- EAN-8, modifier X'00'
- EAN-13, modifier X'00'
- EAN Two-digit supplemental, modifier X'01' (the total height applies to the main symbol; the height of the supplement is calculated from the the main-symbol height)
- EAN Five-digit supplemental, modifier X'01' (the total height applies to the main symbol; the height of the supplement is calculated from the the main-symbol height)

Note: If the total height includes the height of the HRI characters and it is less than or equal to the height of the HRI characters, the result is device dependent. Some BCOCA products report exception condition EC-0700, other products use the total height as the height of the tallest bar.

A value of X'FFFF' indicates the default element height of the presentation device is to be used. For bar code types that explicitly specify the element height, for example, Australia Post Bar Code, Data Matrix, Japan Postal Bar Code, MaxiCode, POSTNET, QR Code, and RM4SCC, this field is ignored. Exception condition EC-0700 exists if the element height specified is invalid or unsupported. For this condition, the bar code object processor uses the closest smaller height. If the smaller value is less than the smallest supported element height or zero, the bar code object processor uses the smallest supported value.

Byte 20 HEIGHT MULTIPLIER

Specifies a value that, when multiplied by the element height, yields the total bar height presented. Exception condition EC-0800 exists if the height multiplier is invalid. For this condition, the bar code object processor uses a height multiplier of X'01'. For bar code types that explicitly specify the height multiplier, for example, Australia Post Bar Code, Data Matrix, Japan Postal Bar Code, MaxiCode, POSTNET, QR Code, and RM4SCC, this field is ignored.

When the default element height (X'FFFF') is specified, the height multiplier value is ignored and a height multiplier of 1 is used.

Bytes 21–22 WE:NE

Specifies the ratio of the wide-element dimension to the narrow-element dimension when only two different size elements exist, that is, for a two-level bar code symbol. The ratio is expressed as a decimal number and normally varies between 2.00 and 3.00.

The WE:NE parameter is used with the following bar code types:
- X'01' – Code 39 (3-of-9 Code), AIM USS-39
- X'02' – MSI (modified Plessey code)
- X'0A' – Industrial 2-of-5
- X'0B' – Matrix 2-of-5
- X'0C' – Interleaved 2-of-5, AIM USS-I 2/5
- X'0D' – Codabar, 2-of-7, AIM USS-Codabar

This parameter is the binary representation of a decimal number of the form n.nnnn; the decimal point follows the first significant digit. For example, a WE:NE value of X'00E1' represents a wide-to-narrow ratio of 2.25 to 1. A particular wide-to-narrow ratio can be encoded in several ways; for example, the WE:NE values X'0015', X'00D2', X'0834', and X'5208' all represent a wide-to-narrow ratio of 2.1 to 1.

The value X'FFFF' indicates that the bar code object processor is to use the default ratio for the specified bar code symbology or presentation device. If the presentation device cannot present the specified narrow-element or wide-element width, exception condition EC-0900 exists. For this condition, the bar code object processor uses the default wide-to-narrow ratio. The default ratio is in the range of 2.25 through 3.00 to 1. The MSI bar code, however, uses a default wide-to-narrow ratio of 2.00 to 1.

The wide-to-narrow ratio parameter is not applicable to the Australia Post Bar Code, Code 93, Code 128, Data Matrix, EAN, Japan Postal Bar Code, MaxiCode, PDF417, POSTNET, QR Code, RM4SCC, and UPC bar code symbologies. The Australia Post Bar Code, Code 93, Data Matrix, Japan Postal Bar Code, MaxiCode, PDF417, POSTNET, QR Code, and RM4SCC symbologies do not define a wide-to-narrow ratio. The Code 128, EAN, and UPC symbologies are referred to as four-level codes. A four-level bar code has four bar-and-space-width levels. The second, third, and fourth levels are automatically calculated as two, three, and four times the module width. When these bar code types are specified, this field is ignored.

Check Digit Calculation Methods

Some bar code types and modifiers call for the calculation and presentation of check digits. Check digits are a method of verifying data integrity during the bar coding reading process. Except for UPC Version E, the check digit is always presented in the bar code bar and space patterns, but is not always presented in the HRI. The following table shows the check digit calculation methods for each bar code type and the presence or absence of the check digit in the HRI.

Bar Code Type	Modifier	In HRI?	Check Digit Calculation
X'01' – Code 39 (3-of-9 Code), AIM USS-39	X'02'	Yes	Modulo 43 of the sum of the data characters' numerical values as described in a Code 39 specification. The start and stop codes are not included in the calculation.
X'02' – MSI (modified Plessey code)	X'02' – X'09'	No	IBM Modulus 10 check digit: 1. Multiply each digit of the original number by a weighting factor of 1 or 2 as follows: multiply the units digit by 2, the tens digit by 1, the hundreds digit by 2, the thousands digit by 1, and so forth. 2. Sum the digits of the products from step 1. This is not the same as summing the values of the products. 3. The check digit is described by the following equation where "sum" is the resulting value of step 2: (10 - (sum modulo 10)) modulo 10
			IBM Modulus 11 check digit: 1. Multiply each digit of the original number by a repeating weighting factor pattern of 2, 3, 4, 5, 6, 7 as follows: multiply the units digit by 2, the tens digit by 3, the hundreds digit by 4, the thousands digit by 5, and so forth. 2. Sum the products from step 1. 3. The check digit depends on the bar code modifier. The check digit as the remainder is described by the following equation where "sum" is the resulting value of step 2: (sum modulo 11) The check digit as 11 minus the remainder is described by the following equation: (11 - (sum modulo 11)) modulo 11

Bar Code Type	Modifier	In HRI?	Check Digit Calculation
			NCR Modulus 11 check digit: 1. Multiply each digit of the original number by a repeating weighting factor pattern of 2, 3, 4, 5, 6, 7, 8, 9 as follows: multiply the units digit by 2, the tens digit by 3, the hundreds digit by 4, the thousands digit by 5, and so forth. 2. Sum the products from step 1. 3. The check digit depends on the bar code modifier. The check digit as the remainder is described by the following equation where "sum" is the resulting value of step 2: (sum modulo 11) The check digit as 11 minus the remainder is described by the following equation: (11 - (sum modulo 11)) modulo 11
X'03' – UPC/CGPC Version A	X'00'	Yes	UPC/EAN check digit calculation: 1. Multiply each digit of the original number by a weighting factor of 1 or 3 as follows: multiply the units digit by 3, the tens digit by 1, the hundreds digit by 3, the thousands digit by 1, and so forth. 2. Sum the products from step 1. 3. The check digit is described by the following equation, where "sum" is the resulting value of step 2: (10 - (sum modulo 10)) modulo 10
X'05' – UPC/CGPC Version E	X'00'	Yes	See X'03' – UPC/CGPC Version A
X'08' – EAN 8 (includes JAN-short)	X'00'	Yes	See X'03' – UPC/CGPC Version A
X'09' – EAN 13 (includes JAN-standard)	X'00'	Yes	See X'03' – UPC/CGPC Version A
X'0A' – Industrial 2-of-5	X'02'	Yes	See X'03' – UPC/CGPC Version A
X'0B' – Matrix 2-of-5	X'02'	Yes	See X'03' – UPC/CGPC Version A
X'0C' – Interleaved 2-of-5	X'02'	Yes	See X'03' – UPC/CGPC Version A
X'0D' – Codabar, 2-of-7, AIM USS-Codabar	X'02'	Varies by receiver	Codabar check digit calculation: 1. Sum of the data characters' numerical values as described in a Codabar specification. All data characters are used, including the start and stop characters. 2. The check digit is described by the following equation where "sum" is the resulting value of step 1: (16 - (sum modulo 16)) modulo 16

Bar Code Type	Modifier	In HRI?	Check Digit Calculation
X'11' – Code 128, AIM USS-128	X'02' – X'03'	No	Code 128 check digit calculation: 1. Going left to right starting at the start character, sum the value of the start character and the weighted values of data and special characters. The weights are 1 for the first data or special character, 2 for the second, 3 for the third, and so forth. The stop character is not included in the calculation. 2. The check digit is modulo 103 of the resulting value of step 1.
X'18' – POSTNET	X'00' – X'04'	NA	The POSTNET check digit is (10 - (sum modulo 10)) modulo 10, where sum is the sum of the user data from the BSA data field.
X'1A' – RM4SCC	X'00'	NA	The RM4SCC checksum digit is calculated using an algorithm that weights each of the 4 bars within a character in relation to its position within the character.
X'1B' – Japan Postal Bar Code	X'00'	NA	The Japan Postal Bar Code check digit calculation: 1. Convert each character in the bar code data into decimal numbers. Numeric characters are converted to decimal, each hyphen character is converted to the number 10, each alphabetic character is converted to two numbers according to the symbology definition. For example, A becomes “11 and 0”, B becomes “11 and 1”, ..., J becomes “11 and 9”, K becomes “12 and 0”, L becomes “12 and 1”, ..., T becomes “12 and 9”, U becomes “13 and 0”, V becomes “13 and 1”, ..., and Z becomes “13 and 5”. 2. Sum the resulting decimal numbers and calculate the remainder modulo 19. 3. The check digit is 19 minus the remainder.
X'1C' – Data Matrix	X'00'	NA	The Data Matrix symbology uses a Reed-Solomon Error Checking and Correcting (ECC) algorithm.
X'1D' – MaxiCode	X'00'	NA	The MaxiCode symbology uses a Reed-Solomon Error Checking and Correcting (ECC) algorithm.
X'1E' – PDF417	X'00' – X'01'	NA	The PDF417 symbology uses a Reed-Solomon Error Checking and Correcting (ECC) algorithm.
X'1F' – Australia Post Bar Code	X'01' – X'08'	No	The Australia Post Bar Code uses a Reed Solomon error correction code based on Galois Field 64.
X'20' — QR Code	X'02'	NA	The QR Code symbology uses a Reed-Solomon Error Checking and Correcting (ECC) algorithm.

Bar Code Type	Modifier	In HRI?	Check Digit Calculation
X'21' — Code 93	X'00'	No	Both check digits (C and K) are calculated as Modulo 47 of the sum of the products of the data-character numerical values as described in the Code 93 specification and a weighting sequence. The start and stop codes are not included in the calculation.

Bar Code Symbol Data (BSA)

The BSA data structure contains the parameters to position the bar code symbol within a bar code presentation space and the data to be encoded. The data is encoded according to the parameters specified in the Bar Code Symbol Descriptor (BSD) data structure.

The format of the BSA data structure follows:

Offset	Type	Name	Range	Meaning	BCD1 Range
0	BITS	Flags			
bit 0		HRI	B'0' B'1'	HRI is presented HRI not presented	B'0' B'1'
bits 1–2		Position	B'00' B'01' B'10'	Default HRI below HRI above	B'00' B'01' B'10'
bit 3		SSCAST	B'0' B'1'	Asterisk is not presented Asterisk is presented	B'0' B'1'
bit 4			B'0'	Reserved	
bit 5		Suppress bar code symbol	B'0' B'1'	Bar code symbol suppression: Present symbol Suppress symbol	B'0'
bit 6		Suppress blanks	B'0' B'1'	Desired method of adjusting for trailing blanks: Don't suppress Suppress and adjust	B'0'
bit 7			B'0'	Reserved	
1–2	UBIN	Xoffset	X'0001'–X'7FFF'	X_{bc}-coordinate of the symbol origin in the bar code presentation space	X'0001'–X'7FFF' Refer to the note following the table.
3–4	UBIN	Yoffset	X'0001'–X'7FFF'	Y_{bc}-coordinate of the symbol origin in the bar code presentation space	X'0001'–X'7FFF' Refer to the note following the table.
The following special-function information is only used with the following bar code types: Data Matrix, MaxiCode, PDF417, QR Code					
5–n		Special functions	See field description	Special-function information that is specific to the bar code type	Not supported in BCD1
The following symbol data is specified for all bar code types					
n+1 to end	UNDF	Data	Any value defined for the bar code type selected by the BSD	Data to be encoded	Any value defined for the bar code type selected by the BSD

Note: The BCD1 range for these fields have been specified assuming a unit of measure of 1/1440 of an inch. Many receivers support the BCD1 subset plus additional function. If a receiver supports additional units of measure, the BCOCA architecture requires the receiver to at least support a range equivalent to the BCD1 range relative to each supported unit of measure. More information about supported-range requirements is provided in the section titled "L-unit Range Conversion Algorithm" on page 20.

The following is a description of the fields defined in the BSA data structure and applicable exception conditions. The standard action to be taken for all exception conditions is to report the exception condition, terminate the bar code object processing, and continue processing with the next object.

Byte 0 Flags

The flags specify attributes specific to this bar code symbol.

The HRI and Position flags indicate the presence and the position of the human-readable interpretation (HRI) of the encoded data. These flags are ignored for symbologies that do not allow HRI or that explicitly specify the presence and position of the HRI; the symbologies for which the HRI flags are ignored include: Data Matrix, Japan Postal Bar Code, MaxiCode, PDF417, POSTNET, QR Code, and RM4SCC.

Bit 0 HRI

If bit 0 is B'0', the HRI is presented.

If bit 0 is B'1', the HRI is not presented.

Bits 1–2
Position

The HRI position flags are used when a bar code symbol and HRI is to be presented. If the bar-code-symbol-suppression flag (bit 5) is B'1', the HRI position flags are ignored and should be set to B'00'.

If bits 1 and 2 are B'00', the presentation device default is used for positioning the HRI.

If bits 1 and 2 are B'01', the HRI is presented below the bar code symbol.

If bits 1 and 2 are B'10', the HRI is presented above the bar code symbol.

If bits 1 and 2 are B'11', exception condition EC-1000 exists.

Note: For the UPC family only, some IPDS printers ignore the position settings and place the HRI as specified in the symbology specification. Specifically, the location of the regular symbol HRI is specified to be below the bars and the supplement symbol HRI above the bars. Other IPDS printers require the position bits to be set according to the symbology specification.

Note: If either the UPC or EAN two-digit and five-digit supplemental bar code is selected in the BSD TYPE field (X'16' or X'17' respectively) and if the BSD MOD (modifier) field has a value other than X'00', the position bits cannot be properly set to indicate the HRI locations for both the regular and supplemental symbol. For these cases, the position bits must be set to the default value setting (B'00').

Bit 3 SSCAST

This flag is used for Code 39 only and is ignored for all other symbologies.

If bit 3 is B'0', no asterisk is presented as the HRI for Code 39 bar code start and stop characters.

If bit 3 is B'1', an asterisk is presented as the HRI for Code 39 bar code start and stop characters.

Bit 4 Reserved

Bit 5 Bar code symbol suppression

This flag specifies whether or not the bar code symbol will be presented, as follows:

B'0' Present the bar code symbol

B'1' Suppress presentation of the bar code symbol. This can be used to print just the HRI. If both bit 0 and bit 5 are B'1' or the bar code does not support HRI, nothing will be presented for this bar code object.

When bit 5 = B'1', the Xoffset and Yoffset parameters specify the character reference point for the first character of the HRI.

Not all BCOCA receivers support suppression of the bar code symbol; receivers that do not support this optional function ignore bit 5.

Bit 6 Desired method of adjusting for trailing blanks

This flag identifies the desired method of handling trailing blanks in the bar code data; for some symbologies, the resulting data length is used to adjust the bar code type and modifier to match the resulting data length.

Note: This flag is used by presentation systems that process AFP line data and may be ignored by BCOCA printers and other presentation systems. AFP line data supports fixed-length fields for bar code data; variable-length fields are not supported. The PAGEDEF formatting-control object that is used with AFP Line Data supports fixed-length fields for data that is to be bar encoded. Since some bar codes allow variable-length data, these fixed-length fields often are padded on the right with blanks; these blanks are often not intended to be included in the BCOCA object, particularly for a bar code type that does not allow blanks. This flag, when specified in a PAGEDEF object, identifies how these trailing blanks should be handled when a BCOCA bar code object is built from the line data and PAGEDEF information.

When AFP line data containing bar code data is processed, this flag is used as follows:

B'0' Do not suppress trailing blanks in the bar code data

B'1' Suppress all trailing blanks in the bar code data and adjust the bar code type and modifier to match the resulting data length.

When the flag = B'1', the bar code data is first adjusted by suppressing trailing blanks and then the bar code type and modifier is adjusted based on the resulting length as follows:

If the user specified an EAN bar code type (X'08', X'09', X'16', or X'17'):

Truncate the data and set the bar code type and modifier based on the resulting data length:

Resulting Data Length	Bar Code Type	Bar Code Modifier
2	X'16' – two-digit supplemental	X'00'
5	X'17' – five-digit supplemental	X'00'
7	X'08' – EAN-8	X'00'
12	X'09' – EAN-13	X'00'
14	X'16' – two-digit supplemental	X'01'
17	X'17' – five-digit supplemental	X'01'
any other value	error	

If the user specified a UPC bar code type (X'03', X'05', X'06', or X'07'):

Truncate the data and set the bar code type and modifier based on the resulting data length:

Resulting Data Length	Bar Code Type	Bar Code Modifier
2	X'06' – two-digit supplemental	X'00'
5	X'07' – five-digit supplemental	X'00'
10	X'05' – UPC version E	X'00'
11	X'03' – UPC version A	X'00'
12	X'06' – two-digit supplemental	X'02'
13	X'06' – two-digit supplemental	X'01'
15	X'07' – five-digit supplemental	X'02'
16	X'07' – five-digit supplemental	X'01'
any other value	error	

If the user specified a POSTNET bar code type (X'18'):

Truncate the data and set the bar code type and modifier based on the resulting data length:

Resulting Data Length	Bar Code Type	Bar Code Modifier
5	X'18' – POSTNET	X'00'
9	X'18' – POSTNET	X'01'
11	X'18' – POSTNET	If X'02' or X'04' was specified, that value is used; if any other modifier was specified, X'02' is used.
any other value	X'18' – POSTNET	X'03'

If the user specified any other bar code type:

Use the user-specified bar code type and modifier.

Bit 7 Reserved

Bytes 1–2 Xoffset

This parameter specifies the origin of the bar code based on the bar code symbol suppression flag (bit 5):

When a bar code symbol is to be presented (bit 5 = B'0'),
this parameter specifies the X_{bc}-coordinate of the top-left corner of the leftmost bar of the bar code symbol. It is referenced to the bar code presentation space origin in the units of measure specified in the BSD data structure.

When a bar code symbol is to be suppressed (bit 5 = B'1'),
this parameter specifies the X_{bc}-coordinate of the character reference point for the first character of the HRI. It is referenced to the bar code presentation space origin in the units of measure specified in the BSD data structure.

Exception condition EC-0A00 exists if the Xoffset value is invalid or unsupported.

Note: For MaxiCode symbols, use the top-left corner of an imaginary rectangle of minimum size that bounds the symbol.

Bytes 3–4 Yoffset

This parameter specifies the origin of the bar code based on the bar code symbol suppression flag (bit 5):

When a bar code symbol is to be presented (bit 5 = B'0'),
this parameter specifies the Y_{bc}-coordinate of the top-left corner of the leftmost bar of the bar code symbol. It is referenced to the bar code presentation space origin in the units of measure specified in the BSD data structure.

When a bar code symbol is to be suppressed (bit 5 = B'1'),
this parameter specifies the Y_{bc}-coordinate of the character reference point for the first character of the HRI. It is referenced to the bar code presentation space origin in the units of measure specified in the BSD data structure.

Exception condition EC-0A00 exists if the Yoffset value is invalid or unsupported.

Note: For MaxiCode symbols, use the top-left corner of an imaginary rectangle of minimum size that bounds the symbol.

Bytes 5–n Special functions specific to the bar code type

The following special-function parameters are only used with the following bar code types, refer to:

"Data Matrix Special-Function Parameters" on page 70
"MaxiCode Special-Function Parameters" on page 75
"PDF417 Special-Function Parameters" on page 81
"QR Code Special-Function Parameters" on page 87

These special-function parameters must not be specified for any other bar code types.

Bytes n+1 to end

Data

Contains the variable data to be encoded and, if required, generated as HRI text characters above or below the bar code symbol. The length and type of data that may be encoded is defined by the bar code symbology. For more information, refer to the appropriate bar code symbology specification listed in Appendix A, "Bar Code Symbology Specification References," on page 109. Exception condition EC-2100 exists if an invalid or undefined character, according to the rules of the bar code symbology specification, is encountered in the bar code data field. Exception condition EC-0C00 exists if the length of the data plus any bar code object processor generated check digit is invalid or unsupported. Refer to Table 8 on page 94 for a description of the valid characters and data length for each symbology.

The data is specified as a series of single-byte code points from a specific code page. Some symbologies limit the valid code points to just the ten numerals (0 through 9), other symbologies allow a richer set of code points. The bar code symbol is produced from these code points; the code points are also used, along with a particular type style, when producing the HRI.

Table 7 on page 93 lists, for each symbology, the valid code page from which characters are chosen and the type style used when printing HRI in terms of an IBM registered CPGID and FGID. More information about these values can be found in *IBM AFP Fonts: Font Summary* and *IBM AFP Fonts: Technical Reference for Code Pages*.

Data Matrix Special-Function Parameters

Offset	Type	Name	Range	Meaning	BCD1 Range
5	BITS			Control flags	
bit 0		EBCDIC	B'0' B'1'	EBCDIC-to-ASCII translation: Do not translate Convert data from EBCDIC to ASCII	Not supported in BCD1
bit 1		Escape sequence handling	B'0' B'1'	Escape-sequence handling: Process escape sequences Ignore all escape sequences	Not supported in BCD1
bits 2-7			B'000000'	Reserved	
6–7	UBIN	Desired row size	X'0000' X'0001'–X'FFFF'	No size specified Matrix row size as allowed by symbology; see field description	Not supported in BCD1
8–9	UBIN	Desired number of rows	X'0000' X'0001'–X'FFFF'	No size specified Number of rows as allowed by symbology; see field description	Not supported in BCD1
10	UBIN	Sequence indicator	X'00'–X'10'	Structured append sequence indicator	Not supported in BCD1
11	UBIN	Total symbols	X'00' or X'02'–X'10'	Total number of structured-append symbols	Not supported in BCD1
12	UBIN	File ID 1st byte	X'01' – X'FE'	High-order byte of a 2-byte unique file identification for a set of structured-append symbols	Not supported in BCD1
13	UBIN	File ID 2nd byte	X'01' – X'FE'	Low-order byte of a 2-byte unique file identification for a set of structured-append symbols	Not supported in BCD1
14	BITS			Special-function flags	
bit 0		UCC/EAN FNC1	B'0' B'1'	Alternate data type identifier: User-defined symbol Symbol conforms to UCC/EAN standards	Not supported in BCD1
bit 1		Industry FNC1	B'0' B'1'	Alternate data type identifier: User-defined symbol Symbol conforms to industry standards	Not supported in BCD1
bit 2		Reader program-ming	B'0' B'1'	Reader programming symbol: Symbol encodes a data symbol Symbol encodes a message used to program the reader system	Not supported in BCD1
bits 3–4		Hdr/Trl Macro	B'00' B'01' B'10' B'11'	Header and trailer instructions to the bar code reader: No header or trailer Use the 05 Macro header/trailer Use the 06 Macro header/trailer No header or trailer	Not supported in BCD1
bits 5–7			B'000'	Reserved	

A desired symbol size can be specified in bytes 6–9, but the actual size of the symbol depends on the amount of data to be encoded. If not enough data is supplied, the symbol will be padded with null data to reach the requested symbol size. If too much data is supplied for the requested symbol size, the symbol will be bigger than requested, but the aspect ratio will be maintained as closely as possible.

Byte 5 Control flags

These flags control how the bar code data (bytes n+1 to end) is processed by the BCOCA receiver; the receiver can be an IPDS printer or any other product that processes BCOCA objects.

Bit 0 EBCDIC-to-ASCII translation

If this flag is B'0', the data is assumed to begin in the default character encodation and no translation is done.

If this flag is B'1', the BCOCA receiver will convert each byte of the bar code data from EBCDIC code page 500 into ASCII code page 819 before this data is used to build the bar code symbol.

Bit 1 Escape-sequence handling

If this flag is B'0', each X'5C' (backslash) within the bar code data is treated as an escape character according to the Data Matrix symbology specification.

If this flag is B'1', each X'5C' within the bar code data is treated as a normal data character and therefore all escape sequences are ignored. In this case, no ECI code page switching can occur within the data.

Note: If the EBCDIC-to-ASCII translation flag is also set to B'1', all EBCDIC backslash characters (X'E0') will first be converted into X'5C' before the escape-sequence handling flag is applied.

Bits 2–7
Reserved

Bytes 6–7 Desired row size

For a Data Matrix symbol, this parameter specifies the desired number of modules in each row including the finder pattern. There must be an even number of modules per row and an even number of rows. There are square symbols with sizes from 10x10 to 144x144, and rectangular symbols with sizes from 8x18 to 16x48 not including quiet zones. The following table lists the complete set of supported sizes. Exception condition EC-0F00 exists if an unsupported size value is specified.

If X'0000' is specified for this parameter, an appropriate row size will be used based on the amount of symbol data.

Table 5. Supported Sizes for a Data Matrix Symbol

Square Symbols				Rectangular Symbols			
Symbol Size		Data Region		Symbol Size		Data Region	
Number of Rows	Row Size	Size	Number	Number of Rows	Row Size	Size	Number
10	10	8x8	1	8	18	6x16	1
12	12	10x10	1	8	32	6x14	2
14	14	12x12	1	12	26	10x24	1
16	16	14x14	1	12	36	10x16	2
18	18	16x16	1	16	36	14x16	2
20	20	18x18	1	16	48	14x22	2
22	22	20x20	1				
24	24	22x22	1				
26	26	24x24	1				
32	32	14x14	4				
36	36	16x16	4				
40	40	18x18	4				
44	44	20x20	4				
48	48	22x22	4				
52	52	24x24	4				
64	64	14x14	16				
72	72	16x16	16				
80	80	18x18	16				
88	88	20x20	16				
96	96	22x22	16				
104	104	24x24	16				
120	120	18x18	36				
132	132	20x20	36				
144	144	22x22	36				

Bytes 8–9 Desired number of rows

For a Data Matrix symbol, this parameter specifies the desired number of rows including the finder pattern. Exception condition EC-0F00 exists if an unsupported size value is specified.

If X'0000' is specified for this parameter, an appropriate number of rows will be used based on the amount of symbol data.

Byte 10 Structured append sequence indicator

Multiple data matrix bar code symbols (called structured appends) can be logically linked together to encode large amounts of data. The logically linked symbols can be presented on the same or on different physical media, and are logically recombined after they are scanned. From 2 to 16 Data Matrix symbols can be linked. This parameter specifies where this symbol is logically linked (1–16) in a sequence of symbols.

If X'00' is specified for this parameter, this symbol is not part of a structured append. Exception condition EC-0F01 exists if an invalid

sequence indicator value is specified. Exception condition EC-0F02 exists if the sequence indicator is larger than the total number of symbols (byte 11).

If this field is not X'00', the reader programming flag must be B'0' and the hdr/trl macro flags must be either B'00' or B'11'. Exception condition EC-0F0A exists if an incompatible combination of these parameters is specified.

Byte 11 Total symbols in a structured append

This parameter specifies the total number of symbols (2–16) that is logically linked in a sequence of symbols.

If X'00' is specified for this parameter, this symbol is not part of a structured append. If this symbol is not part of a structured append, both bytes 10 and 11 must be X'00', or exception condition EC-0F03 exists.

Exception condition EC-0F04 exists if an invalid number of symbols is specified.

Byte 12 High-order byte of structured append file identification

This parameter specifies the high-order byte of a 2-byte unique file identification for a set of structured-append symbols, which helps ensure that the symbols from two different structured appends are not linked together. The low-order byte of the 2-byte field is specified in byte 13. Each of the two bytes can contain a value in the range X'01'–X'FE'.

This parameter is ignored if this symbol is not part of a structured append.

If this symbol is part of a structured append, but byte 12 contains an invalid value (X'00' or X'FF'), exception condition EC-0F0B exists.

Byte 13 Low-order byte of structured append file identification

This parameter specifies the low-order byte of a 2-byte unique file identification for a set of structured-append symbols. The high-order byte of the 2-byte field is specified in byte 12. Each of the two bytes can contain a value in the range X'01'–X'FE'.

This parameter is ignored if this symbol is not part of a structured append.

If this symbol is part of a structured append, but byte 13 contains an invalid value (X'00' or X'FF'), exception condition EC-0F0B exists.

Byte 14 Special-function flags

These flags specify special functions that can be used with a Data Matrix symbol.

Bit 0 UCC/EAN FNC1 alternate data type identifier

If this flag is B'1', an FNC1 shall be added in the first data position (or fifth position of a structured append symbol) to indicate that this symbol conforms to the UCC/EAN application identifier standard format. In this case, the industry FNC1 flag must be B'0', the reader programming

flag must be B'0', and the hdr/trl macro must be B'00' or B'11'. Exception condition EC-0F0A exists if an incompatible combination of these parameters is specified.

Bit 1 Industry FNC1 alternate data type identifier

If this flag is B'1', an FNC1 shall be added in the second data position (or sixth position of a structured append symbol) to indicate that this symbol conforms to a particular industry standard format. In this case, the UCC/EAN FNC1 flag must be B'0', the reader programming flag must be B'0', and the hdr/trl macro must be B'00' or B'11'. Exception condition EC-0F0A exists if an incompatible combination of these parameters is specified.

Bit 2 Reader programming

If this flag is B'1', this symbol encodes a message used to program the reader system. In this case, the structured append sequence indicator must be X'00', the UCC/EAN FNC1 and industry FNC1 flags must both be B'0', and the hdr/trl macro flags must be either B'00' or B'11'. Exception condition EC-0F0A exists if an incompatible combination of these parameters is specified.

Bits 3–4
Header and trailer instructions to the bar code reader

This field provides a means of instructing the bar code reader to insert an industry specific header and trailer around the symbol data.

If this field is B'00' or B'11', no header or trailer is inserted. If this field is B'01', the bar code symbol will contain a 05 Macro codeword. If this field is B'10', the bar code symbol will contain a 06 Macro codeword.

If these flags are B'01' or B'10', the structured append sequence indicator must be X'00', the UCC/EAN FNC1 and industry FNC1 flags must both be B'0', and the reader programming flag must be B'0'. Exception condition EC-0F0A exists if an incompatible combination of these parameters is specified.

Bits 5–7
Reserved

MaxiCode Special-Function Parameters

Offset	Type	Name	Range	Meaning	BCD1 Range
5	BITS			Control flags	
bit 0		EBCDIC	B'0' B'1'	EBCDIC-to-ASCII translation: Do not translate Convert data from EBCDIC to ASCII	Not supported in BCD1
bit 1		Escape sequence handling	B'0' B'1'	Escape-sequence handling: Process escape sequences Ignore all escape sequences	Not supported in BCD1
bits 2–7			B'000000'	Reserved	
6	CODE	Symbol mode	X'02' X'03' X'04' X'05' X'06'	Mode 2 Mode 3 Mode 4 Mode 5 Mode 6	Not supported in BCD1
7	UBIN	Sequence indicator	X'00'–X'08'	Structured append sequence indicator	Not supported in BCD1
8	UBIN	Total symbols	X'00' or X'02'–X'08'	Total number of structured-append symbols	Not supported in BCD1
9	BITS			Special-function flags	
bit 0		Zipper	B'0' B'1'	No zipper pattern Vertical zipper pattern on right	Not supported in BCD1
bits 1–7			B'0000000'	Reserved	

Byte 5 Control flags

These flags control how the bar code data (bytes n+1 to end) is processed by the BCOCA receiver; the receiver can be an IPDS printer or any other product that processes BCOCA objects.

Bit 0 EBCDIC-to-ASCII translation

If this flag is B'0', the data is assumed to begin in the default character encodation and no translation is done.

If this flag is B'1', the BCOCA receiver will convert each byte of the bar code data from EBCDIC code page 500 into ASCII code page 819 before this data is used to build the bar code symbol.

Bit 1 Escape-sequence handling

If this flag is B'0', each X'5C' (backslash) within the bar code data is treated as an escape character according to the MaxiCode symbology specification.

If this flag is B'1', each X'5C' within the bar code data is treated as a normal data character and therefore all escape sequences are ignored. In this case, no ECI code page switching can occur within the data.

Note: If the EBCDIC-to-ASCII translation flag is also set to B'1', all EBCDIC backslash characters (X'E0') will first be converted into X'5C' before the escape-sequence handling flag is applied.

Bits 2–7
Reserved

Byte 6 Symbol mode

Note: The symbol modes are described using the default character encoding (ECI 000003; ASCII code page 819). When the EBCDIC-to-ASCII translation flag is set to B'1', each code point in the data must be specified in EBCDIC. The EBCDIC code point for the “RS” character is X'1E' and the EBCDIC code point for the “GS” character is X'1D'.

Mode 2

Structured Carrier Message - numeric postal code

This mode is designed for use in the transport industry, encoding the postal code, country code, and service class with the postal code being numeric. The bar code data should be structured as described in B.2.1 and B.3.1 of the *AIM International Symbology Specification - MaxiCode*. The postal code, country code, and service class are placed in the primary message portion of the MaxiCode symbol and the rest of the bar code data is placed in the secondary message portion of the MaxiCode symbol. The first part of the bar code data includes the postal code, country code and service class, in that order, separated by the [GS] character (X'1D'). This information may be preceded by the character sequence “[)>RS01GSyy”, where RS and GS are single characters and yy are two decimal digits representing a year. This character sequence represented in hex bytes is X'5B293E1E30311Dxxxx', where each xx is a value from X'30' to X'39'. This sequence indicates that the message conforms to particular open system standards. This first portion of the bar code data must be encoded using the MaxiCode default character set (ECI 000003 = ISO 8859-1). This first portion of the bar code data must not contain the backslash escape character to change the ECI character set. The postal code must be one to nine decimal digits with each digit represented by the byte values from X'30' to X'39'. The country code must be one to three decimal digits with each digit being a byte value from X'30' to X'39'. The service code must also be one to three decimal digits, again with each digit being a byte value from X'30' to X'39'. The primary message portion of the MaxiCode symbol uses Enhanced Error Correction (EEC) and the secondary message portion of the MaxiCode symbol uses Standard Error Correction (SEC).

When the postal code portion of the Structured Carrier Message is numeric, mode 2 should be used.

Mode 3

Structured Carrier Message - alphanumeric postal code

This mode is designed for use in the transport industry, encoding the postal code, country code, and service class with the postal code being alphanumeric. The bar code data should be structured as described in B.2.1 and B.3.1 of the *AIM International Symbology Specification - MaxiCode*. The postal code, country code, and service class are placed in the primary message portion of the MaxiCode symbol

and the rest of the bar code data is placed in the secondary message portion of the MaxiCode symbol. The first part of the bar code data includes the postal code, country code and service class, in that order, separated by the [GS] character (X'1D'). This information may be preceded by the character sequence "[)>RS01GSyy", where RS and GS are single characters and yy are two decimal digits representing a year. This character sequence represented in hex bytes is X'5B293E1E30311Dxxxx', where each xx is a value from X'30' to X'39'. This sequence indicates that the message conforms to particular open system standards. This first portion of the bar code data must be encoded using the MaxiCode default character set (ECI 000003 = ISO 8859-1). This first portion of the bar code data must not contain the backslash escape character to change the ECI character set. The postal code must be one to six alphanumeric characters with each character being one of the printable characters in MaxiCode Code Set A. Postal codes less than 6 characters will be padded with trailing spaces; postal codes longer than 6 characters will be truncated. These characters include the letters A to Z (X'41' to X'5A'), the space character (X'20'), the special characters (X'22' to X'2F'), the decimal digits (X'30' to X'39'), and the colon (X'3A'). The country code must be one to three decimal digits with each digit being a byte value from X'30' to X'39'. The service code must also be one to three decimal digits, again with each digit being a byte value from X'30' to X'39'. The primary message portion of the MaxiCode symbol uses Enhanced Error Correction (EEC) and the secondary message portion of the MaxiCode symbol uses Standard Error Correction (SEC).

When the postal code portion of the Structured Carrier Message is alphanumeric, mode 3 should be used.

Mode 4

Standard Symbol

The symbol employs EEC for the Primary Message and SEC for the Secondary Message. The first nine codewords are placed in the Primary Message and the rest of the codewords are placed in the Secondary Message. This mode provides for a total of 93 codewords for data. If the bar code data consists of only characters from MaxiCode Code Set A, the number of codewords matches the number of bar code data characters. However, if the bar code data contains other characters, the number of codewords is greater than the number of bar code data characters due to the overhead of switching to and from the different code sets. The Code Set A consists of the byte values X'0D', X'1C' to X'1E', X'20', X'22' to X'3A', and X'41' to X'5A'.

Mode 5

Full ECC Symbol

The symbol employs EEC for the Primary Message and EEC for the Secondary Message. The first nine codewords are placed in the Primary Message and the rest of the

codewords are placed in the Secondary Message. This mode provides for a total of 77 codewords for data. If the bar code data consists of only characters from MaxiCode Code Set A, the number of codewords matches the number of bar code data characters. However, if the bar code data contains other characters, the number of codewords is greater than the number of bar code data characters due to the overhead of switching to and from the different code sets. The Code Set A consists of the byte values X'0D', X'1C' to X'1E', X'20', X'22' to X'3A', and X'41' to X'5A'.

Mode 6

Reader Program, SEC

The symbol employs EEC for the Primary Message and SEC for the Secondary Message. The data in the symbol is used to program the bar code reader system. The first nine codewords are placed in the Primary Message and the rest of the codewords are placed in the Secondary Message. This mode provides for a total of 93 codewords for data. If the bar code data consists of only characters from MaxiCode Code Set A, the number of codewords matches the number of bar code data characters. However, if the bar code data contains other characters, the number of codewords is greater than the number of bar code data characters due to the overhead of switching to and from the different code sets. The Code Set A consists of the byte values X'0D', X'1C' to X'1E', X'20', X'22' to X'3A', and X'41' to X'5A'.

Exception condition EC-0F05 exists if an invalid symbol-mode value is specified.

Byte 7 Structured append sequence indicator

Multiple MaxiCode bar code symbols (called structured appends) can be logically linked together to encode large amounts of data. The logically linked symbols can be presented on the same or on different physical media, and are logically recombined after they are scanned. From 2 to 8 MaxiCode symbols can be linked. This parameter specifies where this particular symbol is logically linked (1–8) in a sequence of symbols.

If X'00' is specified for this parameter, this symbol is not part of a structured append. Exception condition EC-0F01 exists if an invalid sequence indicator value is specified. Exception condition EC-0F02 exists if the sequence indicator is larger than the total number of symbols (byte 8).

Byte 8 Total symbols in a structured append

This parameter specifies the total number of symbols (2–8) that is logically linked in a sequence of symbols.

If X'00' is specified for this parameter, this symbol is not part of a structured append. If this symbol is not part of a structured append, both bytes 6 and 7 must be X'00', or exception condition EC-0F03 exists.

Exception condition EC-0F04 exists if an invalid number of symbols is specified.

Byte 9 Special-function flags

These flags specify special functions that can be used with a MaxiCode symbol.

Bit 0 Zipper pattern

If this flag is B'1', a vertical zipper-like test pattern and a contrast block is printed to the right of the symbol. The zipper provides a quick visual check for printing distortions. If the symbol presentation space is rotated, the zipper and contrast block are rotated along with the symbol.

To maintain consistency among printers, the zipper pattern and contrast block should approximate the guideline dimensions shown in Figure 10 on page 80. The zipper pattern and contrast block is made up of several filled rectangles that should be created such that each rectangle is as close to the specified dimensions as possible for the particular printer pel resolution, then the pattern is repeated to yield an evenly spaced zipper pattern and contrast block.

Bits 1–7
Reserved

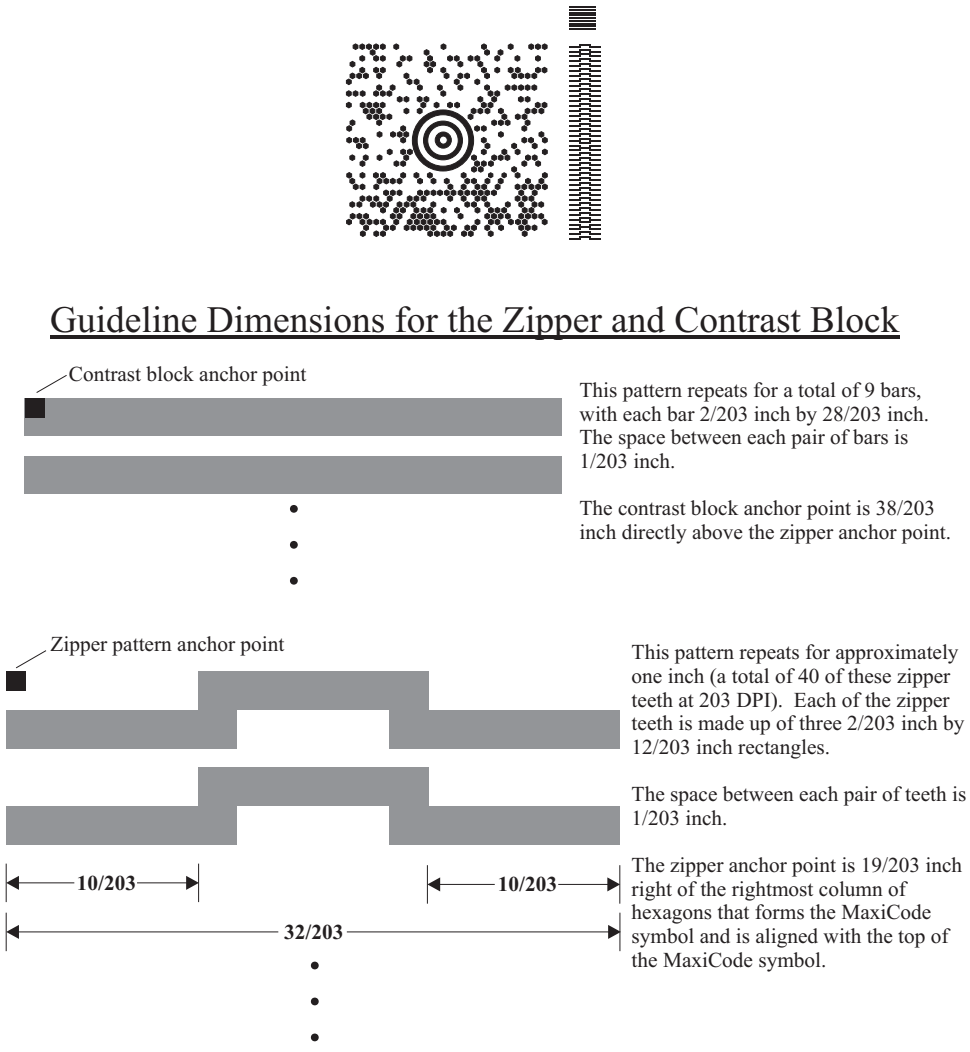


Figure 10. Example of a MaxiCode Bar Code Symbol with Zipper and Contrast Block

PDF417 Special-Function Parameters

Offset	Type	Name	Range	Meaning	BCD1 Range
5	BITS			Control flags	
bit 0		EBCDIC	B'0' B'1'	EBCDIC-to-ASCII translation: Do not translate Convert data from EBCDIC to ASCII	Not supported in BCD1
bit 1		Escape sequence handling	B'0' B'1'	Escape-sequence handling: Process escape sequences Ignore all escape sequences	Not supported in BCD1
bits 2–7			B'000000'	Reserved	
6	UBIN	Data symbols	X'01' – X'1E'	Number of data symbol characters per row	Not supported in BCD1
7	UBIN	Rows	X'03' – X'5A' X'FF'	Desired number of rows Minimum necessary rows	Not supported in BCD1
8	UBIN	Security	X'00' – X'08'	Security level	Not supported in BCD1
9–10	UBIN	Macro length	X'0000'–X'7FED'	Length of Macro PDF417 Control Block that follows	Not supported in BCD1
11–n	UBIN	Macro data	Any value	Data for a Macro PDF417 Control Block	Not supported in BCD1

Byte 5 Control flags

These flags control how the bar code data is processed by the BCOCA receiver; the receiver can be an IPDS printer or any other product that processes BCOCA objects.

Bit 0 EBCDIC-to-ASCII translation (for bytes 11 to end)

If this flag is B'0', the data is assumed to begin in the default character encodation and no translation is done.

If this flag is B'1', the BCOCA receiver will convert each byte of the bar code data (bytes n+1 to end) and each byte of the Macro PDF417 Control Block data (bytes 11–n) from a subset of EBCDIC code page 500 into the default character encodation (GLI 0) before this data is used to build the bar code symbol. This translation covers 181 code points which includes alphanumerics and many symbols; the 75 code points that are not covered by the translation do not occur in EBCDIC and are mapped to X'7F' (127). Refer to Figure 11 on page 82 for a picture showing the 181 EBCDIC code points that can be translated.

The EBCDIC-to-ASCII translation flag should not be used if any of the 75 code points that have no EBCDIC equivalent are needed for the bar code data or for the Macro PDF417 Control Block data.

Table 5 in the Uniform Symbology Specification – PDF417 shows the full set of GLI 0 code points; from this set, the 75 code points that have no EBCDIC equivalent are as follows:

158, 159, 169, 176–224, 226–229, 231–240, 242–245, 247, 249, 251–252, and 254.

The 75 EBCDIC code points that are not covered by the translation and are thus mapped into X'7F' are as follows:

X'04', X'06', X'08'–X'0A', X'14'–X'15', X'17', X'1A'–X'1B', X'20'–X'24', X'28'–X'2C', X'30'–X'31', X'33'–X'36', X'38'–X'3B', X'3E', X'46', X'62', X'64'–X'66', X'6A', X'70', X'72'–X'78', X'80', X'8C'–X'8E', X'9D', X'9F', X'AC'–X'AF', X'B4'–X'B6', X'B9', X'BC'–X'BF', X'CA', X'CF', X'DA', X'EB', X'ED'–X'EF', X'FA'–X'FB', X'FD'–X'FF'.

Hex Digits 1st → 2nd ↓	0-	1-	2-	3-	4-	5-	6-	7-	8-	9-	A-	B-	C-	D-	E-	F-
-0	NUL SE010000	DLE SE170000			(SP) SP010000	& SM030000	_ SP100000			° SM190000	µ SM170000	¢ SC040000	{ SM110000	} SM140000	\ SM070000	0 ND100000
-1	SOH SE020000	DC1 SE180000			(RSP) SP300000	é LE110000	/ SP120000	É LE120000	a LA010000	j LJ010000	~ SD190000	£ SC020000	A LA020000	J LJ020000	÷ SA060000	1 ND010000
-2	STX SE030000	DC2 SE190000		SYN SE230000	â LA150000	ê LE150000			b LB010000	k LK010000	s LS010000	¥ SC050000	B LB020000	K LK020000	S LS020000	2 ND020000
-3	ETX SE040000	DC3 SE200000			ä LA170000	ë LE170000	Ä LA180000		c LC010000	l LL010000	t LT010000	· SD630000	C LC020000	L LL020000	T LT020000	3 ND030000
-4					à LA130000	è LE130000			d LD010000	m LM010000	u LU010000		D LD020000	M LM020000	U LU020000	4 ND040000
-5	HT SE100000		LF SE110000		á LA110000	í LI110000			e LE010000	n LN010000	v LV010000		E LE020000	N LN020000	V LV020000	5 ND050000
-6		BS SE090000	ETB SE240000			î LI150000			f LF010000	o LO010000	w LW010000		F LF020000	O LO020000	W LW020000	6 ND060000
-7	DEL SE330000		ESC SE280000	EOT SE050000	å LA270000	ï LI170000	Å LA280000		g LG010000	p LP010000	x LX010000	¼ NF040000	G LG020000	P LP020000	X LX020000	7 ND070000
-8		CAN SE250000			ç LC410000	ì LI130000	Ç LC420000		h LH010000	q LQ010000	y LY010000	½ NF010000	H LH020000	Q LQ020000	Y LY020000	8 ND080000
-9		EM SE260000			ñ LN190000	ß LS610000	Ñ Ln200000	` SD130000	i LI010000	r LR010000	z LZ010000		I LI020000	R LR020000	Z LZ020000	9 ND090000
-A					[SM060000	] SM080000		: SP130000	« SP170000	ª SM210000	¡ SP030000	¬ SM660000			² ND021000	
-B	VT SE120000				. SP110000	$ SC030000	, SP080000	# SM010000	» SP180000	º SM200000	¿ SP160000	| SM130000	ô LO150000	û LU150000		
-C	FF SE130000	FS SE350000		DC4 SE210000	< SA030000	* SM040000	% SM020000	@ SM050000		æ LA510000			ö LO170000	ü LU170000	Ö LO180000	Ü LU180000
-D	CR SE140000	GS SE360000	ENQ SE060000	NAK SE220000	(SP060000	) SP070000	_ SP090000	' SP050000					ò LO130000	ù LU130000		
-E	SO SE150000	RS SE370000	ACK SE070000		+ SA010000	; SP140000	> SA050000	= SA040000		Æ LA520000			ó LO110000	ú LU110000		
-F	SI SE160000	US SE380000	BEL SE080000	SUB SE270000	! SP020000	^ SD150000	? SP150000	" SP040000	± SA020000					ÿ LY170000		

Figure 11. Subset of EBCDIC Code Page 500 That Can Be Translated To GLI 0

Bit 1 Escape-sequence handling (for bytes n+1 to end)

If this flag is B'0', each X'5C' (backslash) within the bar code data is treated as an escape character according to the PDF417 symbology specification.

If this flag is B'1', each X'5C' within the bar code data is treated as a normal data character and therefore all escape sequences are ignored. In this case, no GLI code page switching and no reader programming can occur within the data.

Note: If the EBCDIC-to-ASCII translation flag is also set to B'1', all EBCDIC backslash characters (X'E0') will first be converted into X'5C' before the escape-sequence handling flag is applied.

Bits 2–7
Reserved

Byte 6 Data symbol characters per row

This parameter specifies the number of data symbol characters per row. Each row consists of a start pattern, a left row indicator codeword, 1 to 30 data symbol characters, a right row indicator codeword (omitted in a truncated symbol), and a stop pattern. The aspect ratio of the bar code symbol is determined by the number of data symbol characters and the number of rows.

Exception condition EC-0F06 exists if an invalid number of data symbol characters per row is specified.

Because of the Error Checking and Correction (ECC) algorithm and the data compaction method used by the printer when the symbol is built, the number of data symbol characters is not necessarily the same as the number of characters in the bar code data.

Byte 7 Desired number of rows

This parameter specifies the desired number of rows in the bar code symbol. From 3 to 90 rows can be specified or X'FF' can be specified to instruct the printer to generate the minimum number of rows necessary. The number of rows times the number of data symbol characters per row cannot exceed 928. Exception condition EC-0F07 exists if an invalid number of rows is specified.

The actual number of rows generated depends on the amount of data to be encoded and on the security level selected. If more rows than necessary are specified, the symbol is padded to fill the requested number of rows. If not enough rows are specified, enough extra rows will be inserted by the printer to produce the symbol.

If too much data is specified to fit in the bar code symbol, exception condition EC-0F08 exists.

Byte 8 Security level

This parameter specifies the desired security level for the symbol as a value between 0 and 8. Each higher security level causes more error correction codewords to be added to the symbol. At a particular security level, a number of codewords can be missing or erased and the symbol can still be recovered. Also, PDF417 can recover from misdecodes of codewords. The formula is: Maximum Limit >= Erasures + 2*Misdecodes The relation of security level to error correction capability is as follows:

Security level	Maximum Limit of Erasures + 2*Misdecodes
0	0
1	2
2	6
3	14
4	30
5	62
6	126
7	254
8	510

For example, at security level 6, a total of 126 codewords can be either missing or destroyed and the entire symbol can still be completely recovered. The following table provides a recommended security level for various amounts of data:

Number of Data Codewords	Recommended Security Level
1–40	2
41–160	3
161–320	4
321–863	5

Exception condition EC-0F09 exists if an invalid security level value is specified.

Bytes 9–10

Length of Macro PDF417 Control Block that follows

This field specifies the length of a Macro PDF417 Control Block that follows in bytes 11–n; this length does not contain the length field itself.

If X'0000' is specified, there is no Macro PDF417 Control Block specified as a special function and this is the last field of the special-function parameters; what follows is the bar code data itself.

If a value between X'0001' and X'7FED' is specified, the BCOCA receiver will build a Macro PDF417 Control Block at the end of the bar code symbol using the data in bytes 11–n.

If an invalid length value is specified, exception condition EC-0F0C exists.

Bytes 11–n

Macro PDF417 Control Block data

The special codewords “\922”, “\923”, and “\928” are used for coding a Macro PDF417 Control Block as defined in section G.2 of the Uniform Symbology Specification PDF417, but these codewords must not be used within the bar code data. Exception condition EC-2100 exists if one of these escape sequences is found in the bar code data. If a Macro PDF417 Control Block is needed, it is specified in bytes 11–n.

The data for this Macro PDF417 Control Block must adhere to the following format; exception condition EC-0F0D exists if this format is not followed:

For the symbol in a Macro PDF417 that represents the last segment of the Macro PDF417, the data must contain “\922”. For all symbols in a Macro PDF417, except the one representing the last segment:

– A Macro PDF417 Control Block starts with a “\928” escape sequence.
– Followed by 1 to 5 numeric digits (bytes values X'30' to X'39'), representing a segment index value from 1 to 99,999.
– Followed by a variable number of escape sequences containing values from “\000” to “\899”, representing the file ID.
– Followed by zero or more optional fields, with the following layout:
  - “\923” escape sequence, signalling an optional field
  - Escape sequence containing the field designator with a value from “\000” to “\006”
  - Followed by a variable number of text characters (for field designators “\000”, “\003”, and “\004”) or a variable number of numeric digits (for field designators “\001”, “\002”, “\005”, and “\006”). The field designators are defined in Table G1 of the Uniform Symbology Specification. For text characters, the byte values must be X'09', X'0A', X'0D', or from X'20' through X'7E'. These values represent the upper case letters A through Z, the lower case letters a through z, and the digits 0 through 9, plus some punctuation and special characters (for GLI 0). For the numeric digits, the byte values must be from X'30' through X'39'.
    • For field designator “\001”, the one to five numeric digits that follow represent the segment count. This value must be greater than or equal to the segment index value.

- For field designator "\002", the one to eleven numeric digits that follow represent the time stamp on the source file expressed as the elapsed time in seconds since January 1, 1970 00:00 GMT.
- For field designator "\005", one or more numeric digits must follow.
- For field designator "\006", the one to five numeric digits that follow represent the decimal value of the 16-bit CRC checksum over the entire source file. This checksum value must be a decimal value from 0 through 65,535.

Note that the file name, segment count, time stamp, sender, addressee, file size, and checksum are provided in the optional fields of the Macro PDF417 Control Block and the BCOCA receiver makes no attempt to calculate or verify these values (other than the previously stated restrictions). If the Macro PDF417 Control Block data does not follow these rules, exception condition EC-0F0D exists. Note that the Uniform Symbology Specification PDF417 has the following additional claims. The BCOCA receiver does not check for these claims nor does it report any exceptions conditions if these claims are violated:

- If the optional Segment Count is given in the Macro PDF417 Control Block of one of the segments (symbols) of the macro, then it should be used in all of the segments (symbols) of the macro.
- All optional fields, other than the Segment Count, only need to appear in one of the segments (symbols) of the macro.
- If an optional field with the same field designator appears in more than one segment (symbol) of the same macro, then it must appear identically in every segment (symbol).

QR Code Special-Function Parameters

Offset	Type	Name	Range	Meaning	BCD1 Range
5	BITS			Control flags	
bit 0		EBCDIC	B'0' B'1'	EBCDIC-to-ASCII translation: Do not translate Convert data to ASCII	Not supported in BCD1
bit 1		Escape sequence handling	B'0' B'1'	Escape-sequence handling: Process escape sequences Ignore all escape sequences	Not supported in BCD1
bits 2–7			B'000000'	Reserved	
6	CODE	EBCDIC code page	X'00' X'01' X'02' X'03'	EBCDIC code page used to encode data: No code page specified Code page 500 (International #5) Code page 290 (Japanese Katakana Ext.) Code page 1027 (Japanese Latin Extended)	Not supported in BCD1
7	CODE	Version	X'00' X'01' – X'28'	Version of symbol: Smallest symbol Version number (1 to 40)	Not supported in BCD1
8	CODE	Error correction level	X'00' X'01' X'02' X'03'	Level of error correction: Level L (7% recovery) Level M (15% recovery) Level Q (25% recovery) Level H (30% recovery)	Not supported in BCD1
9	UBIN	Sequence indicator	X'00' – X'10'	Structured append sequence indicator	Not supported in BCD1
10	UBIN	Total symbols	X'00' or X'02' – X'10'	Total number of structured-append symbols	Not supported in BCD1
11	UBIN	Parity Data	X'00' – X'FF'	Structured append parity data	X'00' – X'FF'
12	BITS			Special-function flags	
bit 0		UCC/EAN FNC1	B'0' B'1'	Alternate data type identifier: User-defined symbol Symbol conforms to UCC/EAN standards	Not supported in BCD1
bit 1		Industry FNC1	B'0' B'1'	Alternate data type identifier: User-defined symbol Symbol conforms to industry standards	Not supported in BCD1
bits 2–7			B'000000'	Reserved	
13	CODE	Application indicator	See field description	Application indicator for Industry FNC1	Not supported in BCD1

A desired symbol size is specified by the version parameter (byte 7), but the actual size of the symbol depends on the amount of data to be encoded. If not enough data is supplied, the symbol will be padded with null data to reach the requested symbol size. If too much data is supplied for the requested symbol size, the symbol will be bigger than requested and will be the smallest symbol that can accommodate that amount of data.

Byte 5 Control flags

These flags control how the bar code data (bytes n+1 to end) is processed by the BCOCA receiver; the receiver can be an IPDS printer or any other product that processes BCOCA objects.

Bit 0 EBCDIC-to-ASCII translation

If this flag is B'0', the data is assumed to begin in the default character encodation (ECI 000020) and no translation is done.

If this flag is B'1' and an EBCDIC code page is selected in byte 6, the BCOCA receiver will convert each byte of the bar code data from the EBCDIC code page specified in byte 6 into ASCII code page 897 before this data is used to build the bar code symbol. The following EBCDIC code pages can be used to encode the bar code data:

- Code page 500 (International #5) — specify X'01' in byte 6

  Only 128 of the characters within ECI 000020 can be specified in code page 500. The code page 500 characters that can be translated are shown in Figure 12 on page 89.
- Code page 290 (Japanese Katakana Extended) — specify X'02' in byte 6
- Code page 1027 (Japanese Latin Extended) — specify X'03' in byte 6

EBCDIC characters that are not defined within ECI 000020 are mapped to X'7F' (DEL).

Bit 1 Escape-sequence handling

If this flag is B'0', each X'5C' (¥) within the bar code data is treated as an escape character according to the QR Code symbology specification.

If this flag is B'1', each X'5C' (¥) within the bar code data is treated as a normal data character and therefore all escape sequences are ignored. In this case, no ECI code page switching can occur within the data.

Note: If the EBCDIC-to-ASCII translation flag is also set to B'1', all EBCDIC ¥ characters will first be converted into X'5C' before the escape-sequence handling flag is applied.

Byte 6 EBCDIC code page

When the EBCDIC-to-ASCII translation flag is B'1', this parameter identifies the EBCDIC code page that encodes the bar code data. The following EBCDIC code pages are supported:

- Code page 500 (International #5) – specify X'01' in byte 6

  Only 128 of the characters within ECI 000020 can be specified in code page 500. The code page 500 characters that can be translated are shown in Figure 12 on page 89.
- Code page 290 (Japanese Katakana Extended) – specify X'02' in byte 6
- Code page 1027 (Japanese Latin Extended) – specify X'03' in byte 6

– If no code page is specified (X'00'), no translation is done.

EBCDIC characters that are not defined within ECI 000020 are mapped to X'7F' (DEL).

When the EBCDIC-to-ASCII translation flag is B'0', this parameter is not used and should be set to X'00'.

Exception condition EC-0F0E exists if an invalid EBCDIC-code-page value is specified.

Hex Digits 1st → 2nd ↓	0-	1-	2-	3-	4-	5-	6-	7-	8-	9-	A-	B-	C-	D-	E-	F-
-0	NUL SE010000	DLE SE170000			(SP) SP010000	& SM030000	_ SP100000						{ SM110000	} SM140000		0 ND100000
-1	SOH SE020000	DC1 SE180000					/ SP120000		a LA010000	j LJ010000			A LA020000	J LJ020000		1 ND010000
-2	STX SE030000	DC2 SE190000		SYN SE230000					b LB010000	k LK010000	s LS010000	¥ SC050000	B LB020000	K LK020000	S LS020000	2 ND020000
-3	ETX SE040000	DC3 SE200000							c LC010000	l LL010000	t LT010000		C LC020000	L LL020000	T LT020000	3 ND030000
-4									d LD010000	m LM010000	u LU010000		D LD020000	M LM020000	U LU020000	4 ND040000
-5	HT SE100000		LF SE110000						e LE010000	n LN010000	v LV010000		E LE020000	N LN020000	V LV020000	5 ND050000
-6		BS SE090000	ETB SE240000						f LF010000	o LO010000	w LW010000		F LF020000	O LO020000	W LW020000	6 ND060000
-7	DEL SE330000		ESC SE280000	EOT SE050000					g LG010000	p LP010000	x LX010000		G LG020000	P LP020000	X LX020000	7 ND070000
-8		CAN SE250000							h LH010000	q LQ010000	y LY010000		H LH020000	Q LQ020000	Y LY020000	8 ND080000
-9		EM SE260000						` SD130000	i LI010000	r LR010000	z LZ010000		I LI020000	R LR020000	Z LZ020000	9 ND090000
-A					[SM060000	] SM080000		: SP130000								
-B	VT SE120000				. SP110000	$ SC030000	, SP080000	# SM010000				| SM130000				
-C	FF SE130000	FS SE350000		DC4 SE210000	< SA030000	* SM040000	% SM020000	@ SM050000				¬ SM150000				
-D	CR SE140000	GS SE360000	ENQ SE060000	NAK SE220000	(SP060000	) SP070000	_ SP090000	' SP050000								
-E	SO SE150000	RS SE370000	ACK SE070000		+ SA010000	; SP140000	> SA050000	= SA040000								
-F	SI SE160000	US SE380000	BEL SE080000	SUB SE270000	! SP020000	^ SD150000	? SP150000	" SP040000								

Figure 12. Subset of EBCDIC Code Page 500 That Can Be Translated To ECI 000020

Byte 7 Version of symbol

This parameter specifies the desired size of the symbol; each version specifies a particular number of modules per row and column. The size of each square module is specified by the module width parameter (byte 17 in the BSD). The following table lists the complete set of supported versions. Exception condition EC-0F0F exists if an invalid version value is specified.

Table 6. Supported Versions for a QR Code Symbol

Version	Symbol Size	Version	Symbol Size
0 (X'00')	smallest	21 (X'15')	101x101
1 (X'01')	21x21	22 (X'16')	105x105
2 (X'02')	25x25	23 (X'17')	109x109
3 (X'03')	29x29	24 (X'18')	113x113
4 (X'04')	33x33	25 (X'19')	117x117
5 (X'05')	37x37	26 (X'1A')	121x121
6 (X'06')	41x41	27 (X'1B')	125x125
7 (X'07')	45x45	28 (X'1C')	129x129
8 (X'08')	49x49	29 (X'1D')	133x133
9 (X'09')	53x53	30 (X'1E')	137x137
10 (X'0A')	57x57	31 (X'1F')	141x141
11 (X'0B')	61x61	32 (X'20')	145x145
12 (X'0C')	65x65	33 (X'21')	149x149
13 (X'0D')	69x69	34 (X'22')	153x153
14 (X'0E')	73x73	35 (X'23')	157x157
15 (X'0F')	77x77	36 (X'24')	161x161
16 (X'10')	81x81	37 (X'25')	165x165
17 (X'11')	85x85	38 (X'26')	169x169
18 (X'12')	89x89	39 (X'27')	173x173
19 (X'13')	93x93	40 (X'28')	177x177
20 (X'14')	97x97		

If X'00' is specified for this parameter, an appropriate row/column size will be used based on the amount of symbol data; the smallest symbol that can accommodate the amount of data is produced.

Byte 8 Level of error correction

This parameter specifies the level of error correction to be used for the symbol. Each higher level of error correction causes more error correction codewords to be added to the symbol and therefore leaves fewer codewords for symbol data. Refer to the QR Code symbology specification for more information about how many codewords are available for symbol data for each version and error-correction level combination.

Four different levels of Reed-Solomon error correction can be selected:
Level L (X'00') allows recovery of 7% of symbol codewords
Level M (X'01') allows recovery of 15% of symbol codewords
Level Q (X'02') allows recovery of 25% of symbol codewords
Level H (X'03') allows recovery of 30% of symbol codewords

Exception condition EC-0F10 exists if an invalid level-of-error-correction value is specified.

Byte 9 Structured append sequence indicator

Multiple QR Code bar code symbols (called structured appends) can be logically linked together to encode large amounts of data. The logically linked symbols can be presented on the same or on different physical media, and are logically recombined after they are scanned. From 2 to 16 QR Code symbols can be linked. This parameter specifies where this symbol is logically linked (1-16) in a sequence of symbols.

If X'00' is specified for this parameter, this symbol is not part of a structured append. Exception condition EC-0F01 exists if an invalid sequence indicator value is specified. Exception condition EC-0F02 exists if the sequence indicator is larger than the total number of symbols (byte 10).

Byte 10
Total number of structured-append symbols

This parameter specifies the total number of symbols (2-16) that is logically linked in a sequence of symbols.

If X'00' is specified for this parameter, this symbol is not part of a structured append. If this symbol is not part of a structured append, both bytes 9 and 10 must be X'00', or exception condition EC-0F03 exists.

Exception condition EC-0F04 exists if an invalid number of symbols is specified.

Byte 11
Structured append parity data

This parameter specifies parity data for a structured append symbol. The parity-data value must be calculated from the entire message that is broken into structured-append symbols; the parity-data value should be the same in each of the structured-append symbols.

The parity-data value is obtained by XORing byte by byte the ASCII/JIS values of all the original input data before division into structured-append symbols.

If this symbol is not a structured append, this parameter is ignored and should be set to X'00'.

Byte 12

Special-function flags

These flags specify special functions that can be used with a QR Code symbol.

Bit 0 UCC/EAN FNC1 alternate data type identifier

If this flag is B'1', this QR Code symbol will indicate that it conforms to the UCC/EAN application identifiers standard. In this case, the industry FNC1 flag must be B'0'. Exception condition EC-0F11 exists if an incompatible combination of these bits is specified.

Bit 1 Industry FNC1 alternate data type identifier

If this flag is B'1', this QR Code symbol will indicate that it conforms to the specific industry or application specifications previously agreed with AIM International. In this case, the UCC/EAN FNC1 flag must be B'0'. Exception condition EC-0F11 exists if an incompatible combination of these bits is specified.

When this flag is B'1', an application indicator is specified in byte 13.

Bits 2–7

Reserved

Byte 13

Application indicator for Industry FNC1

When the Industry FNC1 flag is B'1', this parameter specifies an application indicator. The application indicator is a one-byte value that is specified either as an alphabetic value (from the ASCII set a-z, A-Z) plus 100 or as a two-digit decimal number (between 00 and 99) represented as a hexadecimal value. For example:

for application indicator “a” (ASCII value X'61'), specify X'C5'
for application indicator “Z” (ASCII value X'5A'), specify X'BE'
for application indicator “00”, specify X'00'
for application indicator “01”, specify X'01'
for application indicator “99”, specify X'63'

When the Industry FNC1 flag is B'0', this parameter is ignored and should be set to X'00'.

Exception condition EC-0F12 exists if an invalid application-indicator value is specified.

Valid Code Pages and Type Styles

Table 7. Valid Code Pages and Type Styles

Type	Bar Code Symbology	EBCDIC-Based CPGID	FGID
X'01'	Code 39 (3-of-9 Code), AIM USS-39	500	Device specific
X'02'	MSI (modified Plessey code)	500	Device specific
X'03'	UPC/CGPC — Version A	893	3 (OCR-B)
X'05'	UPC/CGPC — Version E	893	3 (OCR-B)
X'06'	UPC — Two-digit Supplemental (Periodicals)	893	3 (OCR-B)
X'07'	UPC — Five-digit Supplemental (Paperbacks)	893	3 (OCR-B)
X'08'	EAN-8 (includes JAN-short)	893	3 (OCR-B)
X'09'	EAN-13 (includes JAN-standard)	893	3 (OCR-B)
X'0A'	Industrial 2-of-5	500	Device specific
X'0B'	Matrix 2-of-5	500	Device specific
X'0C'	Interleaved 2-of-5, AIM USS-I 2/5	500	Device specific
X'0D'	Codabar, 2-of-7, AIM USS-Codabar	500	Device specific
X'11'	Code 128, AIM USS-128	1303	Device specific
X'16'	EAN Two-digit Supplemental	893	3 (OCR-B)
X'17'	EAN Five-digit Supplemental	893	3 (OCR-B)
X'18'	POSTNET	500	None
X'1A'	RM4SCC	500	None
X'1B'	Japan Postal Bar Code	500	None
X'1C'	Data Matrix (2D bar code)	Code page is selectable within the symbol using ECI protocol	None
X'1D'	MaxiCode (2D bar code)	Code page is selectable within the symbol using ECI protocol	None
X'1E'	PDF417 (2D bar code)	Code page is selectable within the symbol using GLI protocol	None
X'1F'	Australia Post Bar Code	500	Device specific
X'20'	QR Code	Code page is selectable within the symbol using ECI protocol	None
X'21'	Code 93	500	Device specific

Valid Characters and Data Lengths

Table 8 lists the valid characters for each symbology and specifies how many characters are allowed for a bar code symbol. Some bar code symbologies have special rules that identify where in the symbol various characters are allowed. For example, the UPC Version E symbol limits the range of valid values for the last 5 digits based on the value of the first 5 digits. Refer to the appropriate symbology specification for a full description of the rules for laying out bar code data; the symbology specifications are listed in Appendix A, "Bar Code Symbology Specification References," on page 109.

Table 8. Valid Characters and Data Lengths

Code	Bar Code Type	Valid Characters	Valid Data Length
X'01'	Code 39 (3-of-9 Code), AIM USS-39	0123456789 ABCDEFGHIJKLM NOPQRSTUVWXYZ -.$/+% and the space character A total of 43 valid input characters.	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)
X'02'	MSI (modified Plessey code)	0123456789	3 to 15 characters for Modifier X'01' 2 to 14 characters for Modifier X'02' 1 to 13 characters for all other modifiers
X'03'	UPC/CGPC - Version A	0123456789	11 characters
X'05'	UPC/CGPC - Version E	0123456789	10 characters
X'06'	UPC - Two-digit Supplemental (Periodicals)	0123456789	2 characters for Modifier X'00' 13 characters for Modifier X'01' 12 characters for Modifier X'02'
X'07'	UPC - Five-digit Supplemental (Paperbacks)	0123456789	5 characters for Modifier X'00' 16 characters for Modifier X'01' 15 characters for Modifier X'02'
X'08'	EAN-8 (includes JAN-short)	0123456789	7 characters
X'09'	EAN-13 (includes JAN-standard)	0123456789	12 characters
X'0A'	Industrial 2-of-5	0123456789	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)
X'0B'	Matrix 2-of-5	0123456789	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)
X'0C'	Interleaved 2-of-5, AIM USS-I 2/5	0123456789	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)

Table 8. Valid Characters and Data Lengths (continued)

Code	Bar Code Type	Valid Characters	Valid Data Length
X'0D'	Codabar, 2-of-7, AIM USS-Codabar	0123456789 -$:/.+ABCD 16 characters plus 4 start/stop characters (ABCD) (Note 2 on page 97)	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)
X'11'	Code 128, AIM USS-128	All characters defined in the Code 128 code page	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)
X'16'	EAN Two-digit Supplemental	0123456789	2 characters for Modifier X'00' 14 characters for Modifier X'01'
X'17'	EAN Five-digit Supplemental	0123456789	5 characters for Modifier X'00' 17 characters for Modifier X'01'
X'18'	POSTNET	0123456789	5 characters for Modifier X'00' 9 characters for Modifier X'01' 11 characters for Modifier X'02' 11 characters for Modifier X'04' BCOCA range for Modifier X'03': 0 to 50 characters (see note 1 on page 97)
X'1A'	Royal Mail (RM4SCC, modifier X'00')	0123456789 ABCDEFGHIJKLM NOPQRSTUVWXYZ	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)
	Royal Mail (Dutch KIX variation, modifier X'01')	0123456789 ABCDEFGHIJKLM NOPQRSTUVWXYZ abcdefghijklm nopqrstuvwxyz	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)
X'1B'	Japan Postal Bar Code (Modifier X'00')	0123456789 ABCDEFGHIJKLM NOPQRSTUVWXYZ - (hyphen)	Symbology: 7 or more BCOCA range: 7 to 50 characters (see note 1 on page 97)
	Japan Postal Bar Code (Modifier X'01')	0123456789 CC1,CC2,CC3,CC4, CC5,CC6,CC7,CC8 - (hyphen) start,stop	No length checking done; refer to the modifier X'01' description
X'1C'	Data Matrix	Any one-byte character or binary data	Symbology: up to 3116 depending on whether the data is character or numeric; refer to the symbology specification BCOCA range: 0 to 3116 characters (see note 1 on page 97)
X'1D'	MaxiCode	Any one-byte character allowed by the symbol mode; see on page 76	Symbology: up to 93 alphanumeric characters per symbol depending on encoding overhead or up to 138 numeric characters per symbol; refer to the symbology specification BCOCA range: 0 to 138 characters

Table 8. Valid Characters and Data Lengths (continued)

Code	Bar Code Type	Valid Characters	Valid Data Length
X'1E'	PDF417	Any one-byte character or binary data	Symbology: up to 1850 text characters, 2710 ASCII numeric digits, or 1108 bytes of binary data per symbol depending on the security level; refer to the symbology specification BCOCA range: 0 to 2710 characters
X'1F'	**Australia Post Bar Code –**		refer to the modifier (byte 13) description on page 51 to see which characters are valid in specific parts of the symbol
	Modifier X'01' – Standard Customer Barcode	0123456789	Symbology: 8 digits BCOCA range: 8 digits
	Modifier X'02' – Customer Barcode 2 using Table N	0123456789	Symbology: 8–16 digits BCOCA range: 8–16 digits
	Modifier X'03' – Customer Barcode 2 using Table C	0123456789 ABCDEFGHIJKLM NOPQRSTUVWXYZ abcdefghijklm nopqrstuvwxyz (space) # (number sign)	Symbology: 8–13 characters BCOCA range: 8–13 characters
	Modifier X'04' – Customer Barcode 2 using proprietary encoding	0123456789 for sorting code 0–3 for customer information	Symbology: 8–24 digits BCOCA range: 8–24 digits
	Modifier X'05' – Customer Barcode 3 using Table N	0123456789	Symbology: 8–23 digits BCOCA range: 8–23 digits
	Modifier X'06' – Customer Barcode 3 using Table C	0123456789 ABCDEFGHIJKLM NOPQRSTUVWXYZ abcdefghijklm nopqrstuvwxyz (space) # (number sign)	Symbology: 8–18 characters BCOCA range: 8–18 characters
	Modifier X'07' – Customer Barcode 3 using proprietary encoding	0123456789 for sorting code 0–3 for customer information	Symbology: 8–39 digits BCOCA range: 8–39 digits
	Modifier X'08' – Reply Paid Barcode	0123456789	Symbology: 8 digits BCOCA range: 8 digits
X'20'	QR Code	Any one-byte character or binary data	Symbology: Up to 7,089 characters depending on the size and type of the data; refer to the symbology specification BCOCA range: 0 to 7,089 characters
X'21'	Code 93	0123456789 ABCDEFGHIJKLM NOPQRSTUVWXYZ -.$/+% space character a – representing Shift 1 b – representing Shift 2 c – representing Shift 3 d – representing Shift 4 A total of 47 valid input characters.	Symbology: unlimited BCOCA range: 0 to 50 characters (see note 1 on page 97)

Notes:

1. All BCOCA receivers must support at least the BCOCA range. Some receivers support a larger data length.
2. Some descriptions of Codabar show the characters "T,N,*,E" as stop characters (representing the stop characters "A,B,C,D"), but the Codabar symbology actually only allows "A,B,C,D" as start and stop characters. This alternate representation ("T,N,*,E") is used only to distinguish between the start and stop characters when describing a Codabar symbol; when coding a BCOCA Codabar symbol, start and stop characters must be represented using A, B, C, or D.
3. The data for the UPC and EAN symbologies is numeric and of a fixed length, but not all numbers of the appropriate length are valid. This is because the coding scheme is designed to uniquely identify both a product and its manufacturer. The first part of the symbol represents the manufacturer and is defined in the symbology specification (not all numbers are valid in this part of the symbol). The second part of the symbol represents a unique product identifier code assigned by the manufacturer. Refer to the appropriate symbology specification for more details.

Characters and Code Points (Excluding Code 128 and 2D Bar Codes)

Table 9. Characters and Code Points used in the BCOCA Symbologies; Excluding Code 128 and 2D Bar Codes

Character	EBCDIC Code Point	ASCII Code Point
0	X'F0'	X'30'
1	X'F1'	X'31'
2	X'F2'	X'32'
3	X'F3'	X'33'
4	X'F4'	X'34'
5	X'F5'	X'35'
6	X'F6'	X'36'
7	X'F7'	X'37'
8	X'F8'	X'38'
9	X'F9'	X'39'
A	X'C1'	X'41'
B	X'C2'	X'42'
C	X'C3'	X'43'
D	X'C4'	X'44'
E	X'C5'	X'45'
F	X'C6'	X'46'
G	X'C7'	X'47'
H	X'C8'	X'48'
I	X'C9'	X'49'
J	X'D1'	X'4A'
K	X'D2'	X'4B'
L	X'D3'	X'4C'
M	X'D4'	X'4D'
N	X'D5'	X'4E'
O	X'D6'	X'4F'
P	X'D7'	X'50'
Q	X'D8'	X'51'
R	X'D9'	X'52'
S	X'E2'	X'53'
T	X'E3'	X'54'
U	X'E4'	X'55'
V	X'E5'	X'56'
W	X'E6'	X'57'
X	X'E7'	X'58'
Y	X'E8'	X'59'
Z	X'E9'	X'5A'
a	X'81'	X'61'
b	X'82'	X'62'
c	X'83'	X'63'
d	X'84'	X'64'
e	X'85'	X'65'

Table 9. Characters and Code Points used in the BCOCA Symbologies; Excluding Code 128 and 2D Bar Codes (continued)

Character	EBCDIC Code Point	ASCII Code Point
f	X'86'	X'66'
g	X'87'	X'67'
h	X'88'	X'68'
i	X'89'	X'69'
j	X'91'	X'6A'
k	X'92'	X'6B'
l	X'93'	X'6C'
m	X'94'	X'6D'
n	X'95'	X'6E'
o	X'96'	X'6F'
p	X'97'	X'70'
q	X'98'	X'71'
r	X'99'	X'72'
s	X'A2'	X'73'
t	X'A3'	X'74'
u	X'A4'	X'75'
v	X'A5'	X'76'
w	X'A6'	X'77'
x	X'A7'	X'78'
y	X'A8'	X'79'
z	X'A9'	X'7A'
- (hyphen)	X'60'	X'2D'
# (number sign)	X'7B'	X'23'
.	X'4B'	X'2E'
$	X'5B'	X'24'
/	X'61'	X'2F'
+	X'4E'	X'2B'
%	X'6C'	X'25'
:	X'7A'	X'3A'
Space	X'40'	X'20'

Code 128 Code Page

The Code 128 code page (CPGID = 1303, GCSGID = 1454) is defined as shown in Figure 13.

Hex Digits 1st → 2nd ↓	0-	1-	2-	3-	4-	5-	6-	7-	8-	9-	A-	B-	C-	D-	E-	F-
-0	NUL SE010000	DLE SE170000			(SP) SP010000	& SM030000	– SP100000					^ SD150000	{ SM110000	} SM140000	\ SM070000	0 ND100000
-1	SOH SE020000	DC1 SE180000					/ SP120000		a LA010000	j LJ010000	~ SD190000		A LA020000	J LJ020000		1 ND010000
-2	STX SE030000	DC2 SE190000	FS SE350000	SYN SE230000					b LB010000	k LK010000	s LS010000		B LB020000	K LK020000	S LS020000	2 ND020000
-3	ETX SE040000	DC3 SE200000							c LC010000	l LL010000	t LT010000		C LC020000	L LL020000	T LT020000	3 ND030000
-4									d LD010000	m LM010000	u LU010000		D LD020000	M LM020000	U LU020000	4 ND040000
-5	HT SE100000		LF SE110000						e LE010000	n LN010000	v LV010000		E LE020000	N LN020000	V LV020000	5 ND050000
-6		BS SE090000	ETB SE240000						f LF010000	o LO010000	w LW010000		F LF020000	O LO020000	W LW020000	6 ND060000
-7			ESC SE280000	EOT SE050000					g LG010000	p LP010000	x LX010000		G LG020000	P LP020000	X LX020000	7 ND070000
-8		CAN SE250000							h LH010000	q LQ010000	y LY010000		H LH020000	Q LQ020000	Y LY020000	8 ND080000
-9		EM SE260000						` SD130000	i LI010000	r LR010000	z LZ010000		I LI020000	R LR020000	Z LZ020000	9 ND090000
-A						! SP020000		: SP130000				[SM060000			FN2 SE400000	FN3 SE410000
-B	VT SE120000				. SP110000	$ SC030000	, SP080000	# SM010000				] SM080000				
-C	FF SE130000			DC4 SE210000	< SA030000	* SM040000	% SM020000	@ SM050000								
-D	CR SE140000	GS SE360000	ENQ SE060000	NAK SE220000	(SP060000	) SP070000	_ SP090000	' SP050000								
-E	SO SE150000	RS SE370000	ACK SE070000		+ SA010000	; SP140000	> SA050000	= SA040000				FN4 SE420000				
-F	SI SE160000	US SE380000	BEL SE080000	SUB SE270000	| SO130000		? SP150000	" SP040000	FN1 SE390000							DEL SE330000

Note: All START, STOP, SHIFT, and CODE characters are generated by the printer to produce the shortest bar code possible from the given data; these characters are not specified in the Bar Code Symbol Data. All code points not listed in the table are undefined. The code points that do not have graphic character shapes, such as X'00' (NUL) and X'8F' (FN1), are control codes defined within the Code 128 symbology; in the HRI, control codes print in a device-dependent manner. The FN1, FN2, FN3, and FN4 characters are also called FNC 1, FNC 2, FNC 3, and FNC 4 in the Code 128 Symbology Specification.

Figure 13. Code 128 Code Page (CPGID = 1303, GCSGID = 1454)

Chapter 5. Exception Conditions

This chapter lists the BCOCA exception conditions required to be detected by the bar code object processor when processing the bar code data structures and specifies the standard actions to be taken.

Specification-Check Exceptions

A specification-check exception indicates that the bar code object processor has received a bar code request with invalid or unsupported data parameters or values.

Exception
Description

EC-0300
The bar code type specified in the BSD data structure is invalid or unsupported.

Standard Action: Terminate bar code object processing.

EC-0400
A font local ID specified in the BSD data structure is unsupported or not available.

For those symbologies that require a specific type style or code page for HRI, the BCOCA receiver cannot determine the type style or code page of the specified font.

Standard Action: If the requested font is not available, a font substitution can be made preserving as many characteristics as possible of the originally requested font while still preserving the original code page. Otherwise, terminate bar code object processing.

Some bar code symbologies specify a set of type styles to be used for HRI data. Font substitution for HRI data must follow the bar code symbology specification being used.

EC-0500
The color specified in the BSD data structure is invalid or unsupported.

Standard Action: The device default color is used.

EC-0505
The unit base specified in the BSD data structure is invalid or unsupported.

Standard Action: Terminate bar code object processing.

EC-0600
The module width specified in the BSD data structure is invalid or unsupported.

Standard Action: The bar code object processor uses the closest smaller width. If the smaller value is less than the smallest supported width or zero, the bar code object processor uses the smallest supported value.

EC-0605
The units per unit base specified in the BSD data structure is invalid or unsupported.

Standard Action: Terminate bar code object processing.

EC-0700
The element height specified in the BSD data structure is invalid or unsupported.

Standard Action: The bar code object processor uses the closest smaller height. If the smaller value is less than the smallest supported element height or zero, the bar code object processor uses the smallest supported value.

EC-0705
The presentation space extents specified in the BSD data structure are invalid or unsupported.

Standard Action: Terminate bar code object processing.

EC-0800
The height multiplier specified in the BSD data structure is invalid.

Standard Action: The bar code object processor uses X'01'.

EC-0900
The wide-to-narrow ratio specified in the BSD data structure is invalid or unsupported.

Standard Action: The bar code object processor uses the default wide-to-narrow ratio. The default ratio is in the range of 2.25 through 3.00 to 1. The MSI bar code, however, uses a default wide-to-narrow ratio of 2.00 to 1.

EC-0A00
The bar code origin (Xoffset value or Yoffset value) given in the BSA data structure is invalid or unsupported.

Standard Action: Terminate bar code object processing.

EC-0B00
The bar code modifier in the BSD data structure is invalid or unsupported for the bar code type specified in the same BSD.

Standard Action: Terminate bar code object processing.

EC-0C00
The length of the variable data specified in the BSA data structure plus any bar code object processor generated check digits is invalid or unsupported.

Standard Action: Terminate bar code object processing.

EC-0E00
The first check-digit calculation resulted in a value of 10; this is defined as an exception condition in some of the modifier options for MSI bar codes in the BSD data structure.

Standard Action: Terminate bar code object processing.

EC-0F00
Either the matrix row size value or the number of rows value specified in the BSA data structure is unsupported. Both of these values must be within the range of supported sizes for the symbology.

Standard Action: Use X'0000' for the unsupported value so that an appropriate size is used based on the amount of symbol data.

EC-0F01

An invalid structured append sequence indicator was specified in the BSA data structure. For a Data Matrix or QR Code symbol, the sequence indicator must be between 1 and 16 inclusive. For a MaxiCode symbol, the sequence indicator must be between 1 and 8 inclusive.

Standard Action: Present the bar code symbol without structured append information.

EC-0F02

A structured append sequence indicator specified in the BSA data structure is larger than the total number of structured append symbols.

Standard Action: Present the bar code symbol without structured append information.

EC-0F03

Mismatched structured append information was specified in the BSA data structure. One of the sequence-indicator and total-number-of-symbols parameters was X'00', but the other was not X'00'.

Standard Action: Present the bar code symbol without structured append information.

EC-0F04

An invalid number of structured append symbols was specified in the BSA data structure. For a Data Matrix or QR Code symbol, the total number of symbols must be between 2 and 16 inclusive. For a MaxiCode symbol, the total number of symbols must be between 2 and 8 inclusive.

Standard Action: Present the bar code symbol without structured append information.

EC-0F05

For a MaxiCode symbol, the symbol mode value specified in the BSA data structure is invalid.

Standard Action: Terminate bar code object processing.

EC-0F06

For a PDF417 symbol, the number of data symbol characters per row value specified in the BSA data structure is invalid.

Standard Action: Terminate bar code object processing.

EC-0F07

For a PDF417 symbol, the desired number of rows value specified in the BSA data structure is invalid.

This exception condition can also occur when the number of rows times the number of data symbol characters per row is greater than 928.

Standard Action: Proceed as if X'FF' was specified.

EC-0F08

For a PDF417 symbol, too much data was specified in the BSA data structure.

Standard Action: Terminate bar code object processing.

EC-0F09

For a PDF417 symbol, the security level value specified in the BSA data structure is invalid.

Standard Action: Proceed as if security level 8 was specified.

EC-0F0A

An incompatible combination of Data Matrix parameters was specified in the BSA data structure. The following conditions can cause this exception condition:

- A structured append was specified (byte 10 not X'00'), but either the reader programming flag was set to B'1' or a hdr/trl macro was specified.
- The UCC/EAN FNC1 flag was set to B'1', but either the industry FNC1 flag was set to B'1', the reader programming flag was set to B'1', or a hdr/trl macro was specified.
- The industry FNC1 flag was set to B'1', but either the UCC/EAN FNC1 flag was set to B'1', the reader programming flag was set to B'1', or a hdr/trl macro was specified.
- The reader programming flag was set to B'1', but either a structured append was specified, one of the FNC1 flags was set to B'1', or a hdr/trl macro was specified.
- A hdr/trl macro was specified, but either a structured append was specified, one of the FNC1 flags was set to B'1', or the reader programming flag was set to B'1'.

Standard Action: Terminate bar code object processing.

EC-0F0B

An invalid structured append file identification value was specified in the BSA data structure. Each byte of the 2-byte file identification value must be in the range X'01'–X'FE'.

Standard Action: Present the bar code symbol without structured append information.

EC-0F0C

A Macro PDF417 Control Block length value specified in the BSA data structure is invalid.

Standard Action: Terminate bar code object processing.

EC-0F0D

Data within a Macro PDF417 Control Block specified in the BSA data structure is invalid.

Standard Action: Present the bar code symbol without a Macro PDF417 Control Block.

EC-0F0E

For a QR Code symbol, an invalid EBCDIC-code-page value was specified in the BSA data structure.

Standard Action: Terminate bar code object processing.

EC-0F0F

For a QR Code symbol, an invalid version value was specified in the BSA data structure.

Standard Action: Proceed as if X'00' had been specified.

EC-0F10

For a QR Code symbol, an invalid error-correction-level value was specified in the BSA data structure.

Standard Action: Proceed as if X'03' had been specified.

EC-0F11

For a QR Code symbol, an invalid combination of special-function flags was specified in the BSA data structure. Only one of the FNC1 flags can be B'1'.

Standard Action: Terminate bar code object processing.

EC-0F12

For a QR Code symbol, an invalid application-indicator value was specified in the BSA data structure.

Standard Action: Present the bar code symbol without structured append information.

EC-1000

The human-readable interpretation location specified in the the BSA data structure is invalid.

Standard Action: Terminate bar code object processing.

EC-1100

A portion of the bar code, including the bar and space patterns and the HRI, extends outside of either:

- The bar code presentation space
- The intersection of the mapped bar code presentation space and the controlling environment object area
- The maximum presentation area.

Standard Action: Terminate bar code object processing.

All bar code symbols must be presented in their entirety. Whenever a partial bar code pattern is presented, for whatever reason, it is obscured to make it unscannable.

Data-Check Exceptions

A data-check exception indicates that the bar code object processor has detected an undefined character.

Exception
Description

EC-2100

An invalid or undefined character, according to the rules of the symbology specification, has been detected in the bar code data.

Standard Action: Terminate bar code object processing.

Chapter 6. Compliance

This chapter describes compliance rules for generators and receivers of BCOCA data structures.

Generator Rules

A compliant generator is any product that generates semantically and syntactically valid BSD and BSA data structures as defined in Chapter 4, "BCOCA Data Structures," on page 25. For each bar code symbology type to be generated, one and only one BSD can be specified. For each BSD, zero or more BSAs may be defined to generate zero or more bar code symbols of the same type within the bar code presentation space.

Receiver Rules

A compliant receiver is any product that receives and processes BCOCA data structures. A compliant receiver *must*:

- Accept and validate all BCOCA data structure field values defined in the BCD1 range
- Detect and report to the controlling environment all exception conditions as defined in Chapter 5, "Exception Conditions," on page 101
- Support and generate bar code symbols that conform to the bar code symbology specifications listed in Appendix A, "Bar Code Symbology Specification References," on page 109

A compliant receiver *may in addition* accept and validate all BCOCA data structure field values defined in the range.

SAA Compliance

All of the BCOCA data structures are included in SAA Common Communications Support.

Appendix A. Bar Code Symbology Specification References

A general overview and description of most bar code symbologies can be found in the following excellent book. This book also provides information about how to obtain additional bar code symbology information and specifications.

The Bar Code Book written by Roger C. Palmer and published by Helmers Publishing, Inc., 174 Concord Street, Peterborough, New Hampshire 03458. The third edition of this book was published in 1995.

Bar code symbology specifications referred to in this book include:

- *AIM International Technical Specification – International Symbology Specification – Data Matrix*
- *AIM International Technical Specification – International Symbology Specification – MaxiCode*
- *AIM Uniform Symbology Specification – PDF417*
- *AIM International Technical Specification – International Symbology Specification – QR Code*
- Australia Post Bar Code; these publications are available from Australia Post:
  - *Customer Barcoding Technical Specifications*
  - *A Guide to Printing the 4-State Barcode*
- *AIM Uniform Symbology Specification – Code 93*
- *American National Standard Institute For Material Handling (ANSI MH) 10.8M*, American National Standard Institute, New York, NY.
  - Interleaved 2-of-5
  - Industrial 2-of-5
  - Matrix 2-of-5
  - 3-of-9 Code
  - Codabar
- *USS-128 - Uniform Symbology Specification*, Automatic Identification Manufacturers (AIM), Pittsburgh, PA.
  - USS-128 (also known as Code 128)
- *UCC/EAN-128 Application Identifier Standard*, Uniform Code Council, Inc. Dayton, Ohio
  - UCC/EAN 128
- *USS-Codabar - Uniform Symbology Specification*, Automatic Identification Manufacturers (AIM), Pittsburgh, PA.
  - USS-Codabar (also known as Codabar)
- *UPC Symbol Specification Manual*, Uniform Code Council, Dayton, OH.
  - UPC-A, UPC-E, Two-digit Supplemental, Five-digit Supplemental
  - CGPC-A, CGPC-E
- *EAN Symbol Specification Manual*, European Article Numbering Association, Brussels, Belgium.
  - EAN-8, -13, Two-digit Supplemental, Five-Digit Supplemental
- *JIS-STD-X0501*, Japanese Industrial Standards, Japan.
  - JAN-Short, -Standard
- *United States Postal Service Domestic Mail Manual*, United States Printing Office, Washington DC.
  - POSTNET
- *Customer Guide to Confirm using PLANET Codes*, United States Postal Service

  - PLANET Code
- *Bar Code Specification by the Automotive Industry Action Group*, AIAG, Southfield, MI.
  - 3-of-9 Code
  - Interleaved 2-of-5
- American National Standards Institute (Pittsburgh, PA) - Uniform Symbology Specification - Code 39 (August 16, 1995)
- *MIL-STD-1189*, Department of Defense, Philadelphia, PA.
  - 3-of-9 Code
- *Recommended Practices For Uniform Container Symbol/UCS Transport Container Symbol/TCS*, Distribution Symbol Study Group (DSSG), Chicago, IL.
  - USD-1 (Interleaved 2-of-5)
  - USD-2 (3-of-9 Code subset)
- *Uniform Symbol Description*, Material Handling Institute/Automatic Identification Manufacturers Product Section (MHI/AIM), Pittsburgh, PA.
  - USD-1 (Interleaved 2-of-5)
  - USD-2 (3-of-9 Code subset)
  - USD-3 (3-of-9 Code)
  - USD-4 (Codabar, 2-of-7)
  - USD-6 (Code 128)
  - USD-7 (Code 93 - ASCII and non-ASCII versions)
  - USD-8 (Code 11)
- *Bar Code Scanning Reference Guide*, MSI Data Corporation, Costa Mesa, CA.
  - MSI
  - UPC/EAN
  - Code 39
  - Interleaved 2-of-5
- Allais, Dr. David C. *Bar Code Symbology*, Lynnwood, WA: Intermec Corp., 1984.
- *Royal Mail Customer Barcoding Trial Report & Technical Specification*.
  - *KIX Technical Specifications*, PTT Post
- *Japan Postal Bar Code Specification*, available from the Ministry of Postal Service - Japan

Appendix B. MO:DCA Environment

This appendix describes how bar code objects may be included within a MO:DCA-P document for the purpose of interchanging the bar code object between a generating node and one or more receiving nodes. Refer to *Mixed Object Document Content Architecture Reference* for a full description of the MO:DCA architecture.

The description of MO:DCA-P structured fields is included in this appendix solely for setting the context of their use by bar codes.

Bar Codes in MO:DCA-P Documents

The MO:DCA-P bar code object presents one or more bar code symbols of the same type on a page or overlay. Bar code symbols are developed within an abstract bar code presentation space before they are mapped to the MO:DCA-P bar code object area.

The MO:DCA-P Bar Code Data Descriptor (BDD) and Bar Code Data (BDA) structured fields are used to carry bar code object information. These structured fields are described in "Bar Code Data Object Structured Fields" on page 112.

A MO:DCA bar code object has the following basic structure:

Begin Bar Code Object structured field

Object Environment Group (contains the BCOCA BSD structure and other information)

Zero or more **Bar Code Data** structured fields (contains the BCOCA BSA structure); there is one Bar Code Data structured field per bar code symbol

End Bar Code Object structured field

Compliance with MO:DCA-P Interchange Set 2

MO:DCA-P IS/2 is a defined interchange set of the MO:DCA architecture used for presentation documents, that is, documents prepared for presentation on output devices.

To guarantee interchange, a MO:DCA-P IS/2 document carrying a bar code object must include all information related to the object. The MO:DCA-P IS/2 document must, therefore, not only contain the definition of the bar code object, but also provide the linkages to the resources that the object references.

When bar code objects are interchanged with the purpose of outputting objects on a display, printer, or other output device, it is important that visual fidelity be maintained as far as possible. To attempt to satisfy this objective, the BCOCA architecture defines the following for the MO:DCA-P IS/2 environment:

- A set of rules that must be followed by all generators when constructing bar code objects.
- A set of bar code processing capabilities that are guaranteed to be supported by all receivers.

To comply with MO:DCA-P IS/2, products that generate bar code objects must only generate objects that contain bar code data structured fields and values defined in MO:DCA-P IS/2. Including structured fields or values not in the interchange set can result in processing exceptions being raised by the receiving processor, and exception actions taken. However, a generator must not rely on a receiver taking these actions.

Bar Code Data Object Structured Fields

The following sections describe two structured fields: Bar Code Data Descriptor (BDD) and Bar Code Data (BDA).

Bar Code Data Descriptor (BDD)

The BDD specifies the size of the bar code presentation space, the type of bar code to be generated, and the parameters used to generate the bar code symbols.

Structured Field Introducer				
SF Length	SF Identifier X'D3A6EB'	Flags	Sequence Number	Bar Code Symbol Descriptor

The data portion of the BDD structured field is defined in "Bar Code Symbol Descriptor (BSD)" on page 26.

Bar Code Data (BDA)

The BDA structured field contains parameters to position a single bar code symbol within a bar code presentation space, parameters to specify special functions for 2D bar codes, flags to specify attributes specific to the symbol, and the data to be encoded. The data is encoded according to the parameters specified in the Bar Code Data Descriptor (BDD) structured field.

The format of the BDA structured field follows:

Structured Field Introducer				
SF Length	SF Identifier X'D3EEEB'	Flags	Sequence Number	Bar Code Symbol Data

The data portion of the BDA structured field is described in "Bar Code Symbol Data (BSA)" on page 63.

Appendix C. IPDS Environment

Intelligent Printer Data Stream (IPDS) is the IBM data stream for controlling advanced function printer devices. It supports *all points addressable* printing functions that allow text and individual blocks of image, graphics, and bar code data to be positioned and presented at any point on a printed page.

All IPDS printer commands are defined in structured field format that is described in the *Intelligent Printer Data Stream Reference*. Refer to this document for a description of the architecture.

IPDS Bar Code Command Set

The IPDS bar code command set contains the commands and controls for presenting bar code information on a page, a page segment, or an overlay.

The BCOCA bar code object processor is invoked to process the BCOCA data structures contained within the IPDS bar code commands. The BCOCA data structures must contain the BCD1 subset range of field values and may, optionally, contain the full range of field values. The bar code object processor generates the requested bar code symbols on a page, page segment, or overlay.

The IPDS Bar Code Command Set consists of the following commands:
- Write Bar Code Control
- Write Bar Code.

An IPDS bar code object has the following basic structure:

Write Bar Code Control command (contains the BCOCA BSD structure and other information)

Zero or more **Write Bar Code** commands (contains the BCOCA BSA structure); there is one Write Bar Code command per bar code symbol

End command

Write Bar Code Control Command

The Write Bar Code Control command is sent to the printer to direct it to establish initialization parameters for one or more bar code symbols of the same type on the page, page segment, or overlay. The parameters of this command define the bar code presentation space, define the bar code object area, map the bar code presentation space to the bar code object area, and establish the initial conditions for printing the bar code.

The Write Bar Code Control command contains three self-defining fields in the following order:

1. Bar Code Area Position (BCAP) defines the position and orientation of the bar code object area.
2. Bar Code Output Control (BCOC) is optional and specifies the size of the bar code object area, the offset of the presentation space in the bar code object area, and the mapping of the bar code presentation space into the bar code object area.

   The only valid mapping option is *position*. For the position mapping option, the top-left corner of the bar code presentation space, also known as the origin of

the bar code presentation space, is offset from the origin of the bar code object area by the X and Y offset values specified in the BCOC command. If the BCOC is not specified, the origin of the bar code presentation space is coincident with the origin of the bar code object area. Portions of the bar code presentation space may fall outside of the bar code object area without an exception condition being raised. However, exception condition EC-1100 exists if any portion of the bar code, including the bar and space patterns and the HRI, is not totally contained within the bar code object area.

3. Bar Code Data Descriptor (BCDD) defines the bar code presentation space size, the bar code type to be generated, and other associated bar code symbology parameters.

The following defines the format of the BCDD:

Offset	Type	Name	Range	Meaning	BCD1 Range
0–1	UBIN	LENGTH	X'001B' – end of BCDD	Length of BCDD	X'001B' – end of BCDD
2–3	CODE	SDF ID	X'A6EB'	BCDD Self-defining-field ID	X'A6EB'
4–26	UNDF	BSD		Bar Code Symbol Descriptor See "Bar Code Symbol Descriptor (BSD)" on page 26 for parameter definitions.	See "Bar Code Symbol Descriptor (BSD)" on page 26 for BCD1 parameter definitions.

Write Bar Code Command

The Write Bar Code command transmits data to be printed as a single bar code symbol, parameters to specify special functions for 2D bar codes, and flags to specify attributes specific to the symbol. The Write Bar Code command also contains the parameters to position the bar code symbol within the bar code object area. The data portion of the WBC is defined in "Bar Code Symbol Data (BSA)" on page 63.

Additional Related Commands

The following commands are used for query and resource management functions. Only an overview of these commands is presented in this manual. The commands are described in detail in the *Intelligent Printer Data Stream Reference*

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 are also returned as part of the acknowledgement data.

Execute Order Homestate - Obtain Printer Characteristics (XOH OPC): Requests information from the printer that identifies various characteristics of the device. The characteristics include information about the bar code symbologies supported, printable area currently available, coded font resolution, and color support.

Execute Order Anystate - Request Resource List (XOA RRL): Requests the printer to return a specified list of available resources, that is, fonts, overlays, and page

segments, in the Acknowledge Reply to this command. This information can be used by host application programs to perform a variety of resource management functions.

Load Font Equivalence (LFE): This command is sent to the printer to map Local Identifiers referenced in the BCDD to a specific font in the printer.

Font Control Commands: The host can use the following commands to activate and deactivate fonts for printing HRI information:

- Activate Resource
- Load Code Page
- Load Code Page Control
- Load Font
- Load Font Character Set
- Load Font Control
- Load Font Equivalence
- Load Font Index
- Load Symbol Set
- Deactivate Font

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property rights may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2–31 Roppongi 3–chome, Minato-ku
Tokyo 106, Japan

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department 11PA Building 002S
PO Box 1900
Boulder CO 80301 USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee. The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

For online versions of this book, we authorize you to:

- Copy, modify, and print the documentation contained on the media, for use within your enterprise, provided you reproduce the copyright notice, all warning statements, and other required statements on each copy or partial copy.
- Transfer the original unaltered copy of the documentation when you transfer the related IBM product (which may be either machines you own, or programs, if the program's license terms permit a transfer). You must, at the same time, destroy all other copies of the documentation.

You are responsible for payment of any taxes, including personal property taxes, resulting from this authorization.

Your failure to comply with the terms above terminates this authorization. Upon termination, you must destroy your machine readable documentation.

Trademarks

The following terms, used in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

Advanced Function Presentation
Advanced Function Printing
AFP
AIX®
Application System/400®
AS/400®
Bar Code Object Content Architecture
BCOCA
Common User Access®
CUA®
Distributed Relational Database Architecture™
DRDA®
GDDM®
IBM®
ImagePlus®
Infoprint®
Intelligent Printer Data Stream
IPDS
Mixed Object Document Content Architecture
MO:DCA
MVS
MVS/ESA
Operating System/2®
Operating System/400®
OS/2®
OS/400®
Print Services Facility
SAA®

System/370
Systems Application Architecture®
WIN-OS/2®

Microsoft®, Windows®, and Windows NT® are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be the trademarks or service marks of others.

Glossary

This glossary contains terms that apply to the BCOCA Architecture and also terms that apply to other related presentation architectures.

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 *IBM Dictionary of Printing*, document number G544-3973.

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*.

Abstract Syntax Notation One (ASN.1). A notation for defining data structures and data types. The notation is defined in international standard ISO/IEC 8824(E). See also *object identifier*.

ACK. See *Positive Acknowledge Reply*.

Acknowledge Reply. A printer-to-host reply that returns printer information or reports exceptions. An Acknowledge Reply can be positive or negative. See also *Positive Acknowledge Reply* and *Negative Acknowledge Reply*.

Acknowledgment Request. A request from the host for information from the printer. An example of an Acknowledgment Request is the use of the ARQ flag by a host system to request an Acknowledge Reply from an attached printer.

acknowledgment-required flag (ARQ). A flag that requests a printer to return an Acknowledge Reply. The acknowledgment-required flag is bit zero of an IPDS command's flag byte.

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). The IBM strategic environment for presentation.

AEA. See *alternate exception action*.

AFP. See *Advanced Function Presentation*.

AFP data stream. A presentation data stream that is processed in AFP environments. MO:DCA-P is the strategic AFP interchange data stream. IPDS is the strategic AFP printer data stream.

AFPDS. A term formerly used to identify the composed-page MO:DCA-based data stream interchanged in AFP environments. See also *MO:DCA* and *AFP data stream*.

AIAG. See *Automotive Industry Action Group*.

AIM. See *Automatic Identification Manufacturers, Inc*.

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 comment or explanation associated with the contents of a document component. An example of an annotation is a string of text that represents a comment on an image object on a page.

annotation link. In MO:DCA, a link type that specifies the linkage from a source document component to a target document component that contains an annotation.

annotation object. In MO:DCA, an object that contains an annotation. Objects that are targets of annotation links are annotation objects.

ANSI. See *American National Standards Institute*.

APA. See *all points addressable*.

append. In MO:DCA, an addition to or continuation of the contents of a document component. An example of an append is a string of text that is an addition to an existing string of text on a page.

append link. In MO:DCA, a link type that specifies the linkage from the end of a source document component to a target document component that contains an append.

append object. In MO:DCA, an object that contains an append. Objects that are targets of append links are append objects.

application. (1) The use to which an information system is put. (2) The use to which an information system is put. 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.

arc. A continuous portion of the curved line of a circle or ellipse. See also *full arc*.

architected. Identifies data that is defined and controlled by an architecture. Contrast with *unarchitected*.

arc parameters. Variables that specify the curvature of an arc.

area. In GOCA, a set of closed figures that can be filled with a pattern or a color.

area filling. A method used to fill an area with a pattern or a color.

ARQ. See *acknowledgment-required flag*.

array. In FD:OCA, the conceptual model used to describe formatted data. An array describes a string of data fields in terms of dimensions. See also *dimension*.

article. The physical item that a bar code identifies.

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.

ASN.1. See *Abstract Syntax Notation One*.

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. (1) The ratio of the horizontal size of a picture to the vertical size of the picture. (2) In a bar code symbol, the ratio of bar height to symbol length.

asynchronous exception. Any exception other than those used to report a synchronous data-stream defect (action code X'01' or X'1F') or synchronous resource-storage problem (action code X'0C'). Asynchronous exceptions occur after the received page station. An example of an asynchronous exception is a paper jam. See also *data-stream exception*. Contrast with *synchronous exception*.

attribute. A property or characteristic of one or more constructs. See also *character attribute*, *color attribute*, *current drawing attributes*, *default drawing attributes*, *line attributes*, *marker attributes*, and *pattern attributes*.

attribute triplets. In FD:OCA, the part of a descriptor that defines the structure and representation of the data fields.

Automatic Identification Manufacturers, Inc. (AIM). A trade organization consisting of manufacturers, suppliers, and users of bar codes.

Automotive Industry Action Group (AIAG). The coalition of automobile manufacturers and suppliers working to standardize electronic communications within the auto industry.

B

b_c. See *current baseline print coordinate*.

b_i. See *initial baseline print coordinate*.

B. See *baseline direction*.

+B. Positive baseline direction.

B_c. See *current baseline presentation coordinate*.

B_o. See *baseline presentation origin*.

background.

1. The part of a presentation space that is not occupied with object data.
2. In GOCA, that portion of a graphics primitive that is mixed into the presentation space under the control of the current values of the background mix and background color attributes. Contrast with *foreground*.
3. In GOCA, that portion of a character cell that does not represent a character.
4. In bar codes, the spaces, quiet zones, and area surrounding a printed bar code symbol.

background color. The color of a background. Contrast with *foreground color*.

background mix. (1) An attribute that determines how the color of the background of a graphics primitive is combined with the existing color of the graphics presentation space. (2) An attribute that determines how the points in overlapping presentation space backgrounds are combined. Contrast with *foreground mix*.

band. An arbitrary layer of an image. An image can consist of one or more bands of data.

bar. In bar codes, the darker element of a printed bar code symbol. See also *element*. Contrast with *space*.

bar code. An array of elements, such as bars, spaces, and two-dimensional modules that together represent data elements or characters in a particular symbology. The elements are arranged in a predetermined pattern following unambiguous rules defined by the symbology. See also *bar code symbol*.

bar code command set. In the IPDS architecture, a collection of commands used to present bar code symbols in a page, page segment, or overlay.

bar code density. The number of characters per inch (cpi) in a bar code symbology. In most cases, the range is three to ten cpi. See also *character density*, *density*, and *information density*.

bar code object area. The rectangular area on a logical page into which a bar code presentation space is mapped.

Bar Code Object Content Architecture (BCOCA). An architected collection of constructs used to interchange and present bar code data.

bar code presentation space. A two-dimensional conceptual space in which bar code symbols are generated.

bar code symbol. A combination of characters including start and stop characters, quiet zones, data characters, and check characters required by a particular symbology, that form a complete, scannable entity. See also *bar code*.

bar code symbology. A bar code language. Bar code symbologies are defined and controlled by various industry groups and standards organizations. Bar code symbologies are described in public domain bar code specification documents. Synonymous with *symbology*. See also *Canadian Grocery Product Code (CGPC)*, *European Article Numbering (EAN)*, *Japanese Article Numbering (JAN)*, and *Universal Product Code (UPC)*.

bar height. In bar codes, the bar dimension perpendicular to the bar width. Synonymous with *bar length* and *height*.

bar length. In bar codes, the bar dimension perpendicular to the bar width. Synonymous with *bar height* and *height*.

bar width. In bar codes, the thickness of a bar measured from the edge closest to the symbol start character to the trailing edge of the same bar.

bar width reduction. In bar codes, the reduction of the nominal bar width dimension on film masters or printing plates to compensate for systematic errors in some printing processes.

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*.

Begin Segment Introducer (BSI). An IPDS graphics self-defining field that precedes all of the drawing orders in a graphics segment.

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.

BITS. A data type for architecture syntax, indicating one or more bytes to be interpreted as bit string information.

body. (1) On a printed page, the area between the top and bottom margins that can contain data. (2) In a book, the portion between the front matter and the back matter.

boldface. A heavy-faced type. Printing in a heavy-faced type.

boundary alignment. A method used to align image data elements by adding padding bits to each image data element.

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.

BSI. See *Begin Segment Introducer*.

B-space. The distance between the character coordinate system X-axis values of the two extremities of a character shape. See also *A-space* and *C-space*.

buffered pages. Pages and copies of pages that have been received but not yet reflected in committed page counters and copy counters.

C

called segment. A segment that is called from another segment. It can be regarded as an extension of the calling segment, but some actions take place at the call and others at the return. Examples of actions are saving the addresses of the current position and the next order on the segment call stack at the call and restoring those saved addresses at the return. See also *segment call stack*.

Canadian Grocery Product Code (CGPC). The bar code symbology used to code grocery items in Canada.

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*.

CCS. See *Common Communications Support*.

CCSID. See *Coded Character Set Identifier*.

CGCSGID. See *Coded Graphic Character Set Global Identifier*.

CGPC. See *Canadian Grocery Product Code*.

CHAR. A data type for architecture syntax, indicating one or more bytes to be interpreted as character information.

character. (1) 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*. (2) In bar codes, a single group of bar code elements that represent an individual number, letter, punctuation mark, or other symbol.

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 density. The number of characters per inch (cpi) in a bar code symbology. In most cases, the range is three to ten cpi. See also *bar code density*, *density*, and *information density*.

character direction. In GOCA, an attribute controlling the direction in which a character string grows relative to the inline direction. Values are: left-to-right, right-to-left, top-to-bottom, and bottom-to-top. Synonymous with *direction*.

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 coded font.

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. Character rotation and font inline sequence are related in that character rotation is a clockwise rotation; font inline sequence is a counter-clockwise rotation. 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 coded 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.

check character. In bar codes, a character included within a bar code message whose value is used to perform a mathematical check to ensure the accuracy of that message. Synonymous with *check digit*.

check digit. In bar codes, a character included within a bar code message whose value is used to perform a mathematical check to ensure the accuracy of that message. Synonymous with *check character*.

clear area. A clear space that contains no machine-readable marks preceding the start character of a bar code symbol or following the stop character. Synonymous with *quiet zone*. Contrast with *intercharacter gap* and *space*.

clipping. Eliminating those parts of a picture that are outside of a clipping boundary such as a viewing window or presentation space. See also *viewing window*. Synonymous with *trimming*.

Codabar. A bar code symbology characterized by a discrete, self-checking, numeric code with each character represented by a stand-alone group of four bars and the three spaces between them.

CODE. A data type for architecture syntax that indicates an architected constant to be interpreted as defined by the architecture.

Code 39. A bar code symbology characterized by a variable-length, bidirectional, discrete, self-checking, alphanumeric code. Three of the nine elements are wide and six are narrow. It is the standard for LOGMARS (the Department of Defense) and the AIAG.

Code 128. A bar code symbology characterized by a variable-length, alphanumeric code with 128 characters.

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 that are directly paired to font metrics and the raster representation of character shapes, for a specific graphic character set. (4) In the IPDS architecture, a font resource containing descriptive information, a code page, font metrics, and a digital-technology representation of character shapes for a specific graphic character set.

coded 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. Graphic characters can be added over time; therefore, to specifically identify a code page, both a GCSGID and a CPGID should be used. See also *coded graphic character set*. (2) A set of assignments, each of which assigns a code point to a character. Each code page has a unique name or identifier. Within a given code page, a code point is assigned to one character. More than one character set can be assigned code points from the same code page. See also *code point* and *section*.

Code Page Global Identifier (CPGID). A unique code page identifier that can be expressed as either a two-byte binary or a five-digit decimal value.

code point. A unique bit pattern that can serve as an element of a code page or a site in a code table, to which a character can be assigned. The element is associated with a binary value. The assignment of a character to an element of a code page determines the binary value that will be used to represent each occurrence of the character in a character string. Code points are one or more bytes long. See also *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 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 image. Images whose image data elements are represented by multiple bits or whose image data element values are mapped to color values. Constructs that map image-data-element values to color values are look-up tables and image-data-element structure parameters. Examples of color values are screen color values for displays and color toner values for printers.

color model. 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). Also referred to as *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). Also referred to as *color model*.

color table. A collection of color element sets. The table can also specify the method used to combine the intensity levels of each element in an element set to produce a specific color. Examples of methods used to combine intensity levels are the additive method and the subtractive method. See also *color model*.

column. In FD:OCA, a subarray consisting of all elements that have an identical position within the low dimension of a regular two-dimensional array.

command. (1) In the IPDS architecture, a structured field sent from a host to a printer. (2) In GOCA, a data-stream construct used to communicate from the controlling environment to the drawing process. (3) The command introducer is environment dependent. A request for system action.

command set. A collection of IPDS commands.

command-set vector. Information that identifies an IPDS command set and data level supported by a printer. Command-set vectors are returned with an Acknowledge Reply to an IPDS Sense Type and Model command.

Common Communications Support (CCS). Protocols and conventions for connecting systems and software. One of the three SAA architectural areas (the other two being Common Programming Interface and Common User Access).

CPI. See *Common Programming Interface*.

Common Programming Interface (CPI). Definitions of those application development languages and services that have (or are intended to have) implementations on and a high degree of commonality across the SAA environments. One of the three SAA architectural areas (the other two being Common Communications Support and Common User Access).

Common User Access (CUA). Guidelines for the dialog between a person and the workstation or terminal. One of the three SAA architectural areas (the other two being Common Programming Interface and Common Communications Support).

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 code. A bar code symbology characterized by designating all spaces within the symbol as parts of characters, for example, Interleaved 2 of 5. There is no intercharacter gap in a continuous code. Contrast with *discrete code*.

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. (1) A character that denotes the start, modification, or end of a control function. A control character can be recorded for use in a subsequent action, and it can have a graphic representation. See also *character*. (2) A control function the coded representation of which consists of a single code point.

control instruction. A data construct transmitted from the controlling environment and interpreted by the environment interface to control the operation of the graphics processor.

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.

control sequence. A sequence of bytes that specifies a control function. A control sequence consists of a control sequence introducer and zero or more parameter s.

control sequence chaining. A method used to identify a sequential string of control sequence s 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'.

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*.

copy control. A method used to specify the number of copies for a presentation space and the modifications to be made to each copy.

copy counter. Bytes in an Acknowledge Reply that identify the number of copies of a page that have passed a particular point in the logical paper path.

copy group. A set of copy subgroups that specify all copies of a sheet. In the IPDS architecture, a copy group is specified by a Load Copy Control command. In MO:DCA, a copy group is specified within a Medium Map. See also *copy subgroup*.

copy modification. The process of adding, deleting, or replacing data on selected copies of a presentation space.

copy subgroup. A part of a copy group that specifies a number of identical copies of a sheet and all modifications to those copies. Modifications include the media source, the media destination, medium overlays to be presented on the sheet, text suppressions, the number of pages on the sheet, and either simplex or duplex presentation. In the IPDS architecture, copy subgroups are specified by Load Copy Control command entries. In MO:DCA, copy subgroups are specified by repeating groups in the Medium Copy Count structured field in a Medium Map. See also *copy group*.

correlation. (1) A method used in GOCA to determine if a picture defines any parts of a drawing that lie within a pick window. See also *pick window*. (2) A method used in the IPDS architecture to match exceptions with commands.

correlation ID. A two-byte value that specifies an identifier of an IPDS command. The correlation ID is optional and is present only if bit one of the command's flag byte is B'1'.

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*.

CUA. See *Common User Access*.

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). 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 coordinate*.

current baseline print coordinate (b_c). In the IPDS architecture, the baseline coordinate corresponding to the current print position on a logical page. The current baseline print coordinate is a coordinate in an I,B coordinate system. See also *I,B coordinate system*.

current drawing attributes. The set of attributes used at the present time to direct a drawing process. Contrast with *default drawing attributes*.

current drawing controls. The set of drawing controls used at the present time to direct a drawing process. Contrast with *default drawing controls*.

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). 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 coordinate*.

current inline print coordinate (i_c). In the IPDS architecture, the inline coordinate corresponding to the current print position on a logical page. The current inline print coordinate is a coordinate in an I,B coordinate system. See also *I,B coordinate system*.

current logical page. The logical page presentation space that is currently being used to process the data within a page object or an overlay object.

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 block. A deprecated term for object area.

data element. A unit of data that is considered indivisible.

data frame. A rectangular division of computer output on microfilm.

data 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.

data-stream exception. In the IPDS architecture, a condition that exists when the printer detects an invalid or unsupported command, order, control, or parameter value from the host. Data-stream exceptions are those whose action code is X'01', X'19', or X'1F'. See also *asynchronous exception* and *synchronous exception*.

DBCS. See *double-byte character set*.

decoder. In bar codes, the component of a bar code reading system that receives the signals from the scanner, performs the algorithm to interpret the signals into meaningful data, and provides the interface to other devices. See also *reader* and *scanner*.

default. A value, attribute, or option that is assumed when none has been specified and one is needed to continue processing. See also *default drawing attributes* and *default drawing controls*.

default drawing attributes. The set of drawing attributes adopted at the beginning of a drawing process and usually at the beginning of each root segment that is processed. See also *root segment*. Contrast with *current drawing attributes*.

default drawing controls. The set of drawing controls adopted at the start of a drawing process and usually at the start of each root segment that is processed. See also *root segment*. Contrast with *current drawing controls*.

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*.

density. The number of characters per inch (cpi) in a bar code symbology. In most cases, the range is three to ten cpi. See also *bar code density*, *character density*, and *information density*.

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-control command set. In the IPDS architecture, a collection of commands used to set up a page, communicate device controls, and manage printer acknowledgment protocol.

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.

device-version code page. In the IPDS architecture, a device version of a code page contains all of the characters that were registered for the CPGID at the time the printer was developed; since then, more characters might have been added to the registry for that CPGID. A device-version code page is identified by a CPGID. See also *code page*.

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.

dimension. In FD:OCA, each successive level of partitioning. Dimensions allow the addressing of specific parts of an array. See also *partitioning* and *array*.

direction. In GOCA, an attribute that controls the direction in which a character string grows relative to the inline direction. Values are: left-to-right, right-to-left, top-to-bottom, and bottom-to-top. Synonymous with *character direction*.

discrete code. A bar code symbology characterized by placing spaces that are not a part of the code between characters, that is, intercharacter gaps.

Distributed Relational Database Architecture (DRDA). A protocol that allows applications to access data from remote databases.

DOCS. See *drawing order coordinate space*.

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, which 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 65536 characters.

double-byte coded font. A coded font in which the code points are two bytes long.

downloaded resource. In the IPDS architecture, a resource in a printer that is installed and removed under control of a host presentation services program. A downloaded resource is referenced by a host-assigned name that is valid for the duration of the session between the presentation services program and the printer. Contrast with *resident resource*.

drag. To use a pointing device to move an object. For example, clicking on a window border, and dragging it to make the window larger.

draw functions. Functions that can be done during the drawing of a picture. Examples of draw functions are displaying a picture, correlation, boundary computation, and erasing a graphics presentation space.

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.

drawing control. A control that determines how a picture is drawn. Examples of drawing controls are arc parameters, transforms, and the viewing window.

drawing order. In GOCA, a graphics construct that the controlling environment builds to instruct a drawing processor about what to draw and how to draw it. The order can specify, for example, that a graphics primitive be drawn, a change to drawing attributes or drawing controls be effected, or a segment be called. One or more graphics primitives can be used to draw a picture. Drawing orders can be included in a structured field. See also *order*.

drawing order coordinate space (DOCS). A two-dimensional conceptual space in which graphics primitives are drawn, using drawing orders, to create pictures.

drawing processor. A graphics processor component that executes segments to draw a picture in a presentation space. See also *segment*, *graphics presentation space*, and *image presentation space*.

drawing units. Units of measurement used within a graphics presentation space to specify absolute and relative positions.

DRDA. See *Distributed Relational Database Architecture*.

duplex. A method used to print data on both sides of a sheet. Normal-duplex printing occurs when the sheet is turned over the Y_m axis. Tumble-duplex printing occurs when the sheet is turned over the X_m axis.

duplex printing. A method used to print data on both sides of a sheet. Contrast with *simplex printing*.

dynamic segment. A segment whose graphics primitives can be redrawn in different positions by dragging them from one position to the next across a picture without destroying the traversed parts of the picture.

E

EAN. See *European Article Numbering*.

EBCDIC. See *Extended Binary-Coded Decimal Interchange Code*.

element.

1. A bar or space in a bar code character or a bar code symbol.
2. A structured field in a document content architecture data stream.
3. In GOCA, a portion of a segment consisting of either a single order or a group of orders enclosed in an element bracket, in other words, between a *begin* element and an *end* element.
4. In FD:OCA, each of the data fields in an array.
5. A basic member of a mathematical or logical class or set.

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.

Em square. A square layout space used for designing each of the characters of a font.

encoding scheme. A set of specific definitions that describe the philosophy used to represent character data. The number of bits, the number of bytes, the allowable ranges of bytes, the maximum number of characters, and the meanings assigned to some generic and specific bit patterns, are some examples of specifications to be found in such a definition.

Encoding Scheme Identifier (ESID). A 16-bit number assigned to uniquely identify a particular encoding scheme specification. See also *encoding scheme*.

environment interface. The part of the graphics processor that interprets commands and instructions from the controlling environment.

escape sequence.

1. In the IPDS architecture, the first two bytes of a control sequence. An example of an escape sequence is X'2BD3'.
2. 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.

European Article Numbering (EAN). The bar code symbology used to code grocery items in Europe.

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.

See also *data-stream exception*, *asynchronous exception*, and *synchronous exception*.

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.

extent. In FD:OCA, one of the characteristics of a dimension. If all partitions of a dimension have the same number of subpartitions, then this number is called the extent of the next lower dimension. See also *local extent*.

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

factoring. The movement of a parameter value from one state to a higher-level state. This permits the parameter value to apply to all of the lower-level states unless specifically overridden at the lower level.

FDO. See *formatted data object*.

FD:OCA. See *Formatted Data Object Content Architecture*.

FGID. See *Font Typeface Global Identifier*.

fillet. A curved line drawn tangential to a specified set of straight lines. An example of a fillet is the concave junction formed where two lines meet.

final form data. Data that has been formatted for presentation.

first read rate. In bar codes, the ratio of the number of successful reads on the first attempt to the total number of attempts made to obtain a successful read. Synonymous with *read rate*.

fixed medium information. Information that can be applied to a sheet by a printer or printer-attached device that is independent of data provided through the data stream. Fixed medium information does not mix with the data provided by the data stream and is presented on a sheet either before or after the text, image, graphics, or bar code data provided within the data stream. Fixed medium information can be used to create "pre-printed forms", or other types of printing, such as colored logos or letterheads, that cannot be created conveniently within the data stream.

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 control record. The record sent in an IPDS Load Font Control command to specify a font ID and other font parameters that apply to the complete font.

Font Typeface Global Identifier (FGID). A unique font identifier that can be expressed as either a two-byte binary or a five-digit decimal value. The FGID is used to identify a type style and the following characteristics: posture, weight, and width.

font height (FH).

1. A characteristic value, perpendicular to the character baseline, that represents the size of all graphic characters in a font. Synonymous with *vertical font size*.
2. In a font character set, nominal font height 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 font height 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.

font index.

1. The mapping of a descriptive font name to a font member name in a font library. An example of a font member in a font library is a font resource object. Examples of attributes used to form a descriptive font name are typeface, family name, point size, style, weight, and width.
2. In the IPDS architecture, an LF1-type raster-font resource containing character metrics for each code point of a raster font or raster-font section for a particular font inline sequence. There can be a font index for 0 degree, 90 degree, 180 degree, and 270 degree font inline sequences. A font index can be downloaded to a printer using the Load Font Index command. An LF1-type coded font or coded-font section is the combination of one fully described font and one font index. See also *fully described font*.

font inline sequence. The clockwise rotation of the inline direction relative to a character pattern. Character rotation and font inline sequence are related in that character rotation is a clockwise rotation; font inline sequence is a counter-clockwise rotation.

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 order s, 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).

1. A characteristic value, parallel to the character baseline, that represents the size of all graphic characters in a font. Synonymous with *horizontal font size*.
2. In a font character set, nominal font width is a font-designer defined value corresponding to the nominal character increment for a font character set. The value is generally the width of the space character and is defined differently for fonts with different spacing characteristics.

- For fixed-pitch, uniform character increment fonts: the fixed character increment, which is also the space character increment
- For PSM fonts: the width of the space character
- For typographic, proportionally spaced fonts: one-third of the vertical font size, which is also the default size of the space character.

The font designer can also define a minimum and a maximum horizontal font size to represent the limits of scaling.

3. In font referencing, the specified font width is the desired size of the font when the characters are presented. If this size is different from the nominal horizontal font size specified in a font character set, the character shapes and character metrics might need to be scaled prior to presentation.

foreground.

1. The part of a presentation space that is occupied by object data.
2. In GOCA, the portion of a drawing primitive that is mixed into the presentation space under the control of the current value of the mix and color attributes. See also *pel*. Contrast with *background*.

foreground color. A color attribute used to specify the color of the foreground of a primitive. Contrast with *background color*.

foreground mix. An attribute 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*.

form. A division of the physical medium; multiple forms can exist on a physical medium. For example, a roll of paper might be divided by a printer into rectangular pieces of paper, each representing a form. Envelopes are an example of a physical medium that comprises only one form. The IPDS architecture defines four types of forms: cut-sheets, continuous forms, envelopes, and computer output on microfilm. Each type of form has a top edge. A form has two sides, a front side and a back side. Synonymous with *sheet*.

format. The arrangement or layout of data on a physical medium or in a presentation space.

formatted data. In FD:OCA, data whose implied syntax and semantics are represented by architected controls that accompany the data.

formatted data object (FDO). An object that contains formatted data. See also *object*.

Formatted Data Object Content Architecture (FD:OCA). An architected collection of constructs used to interchange formatted data.

formatter. A process used to prepare a document for presentation.

Formdef. See *Form Definition*.

Form Definition (Formdef). A print control object that contains an environment definition and one or more Medium Maps. Synonymous with *Form map*.

Form Map. A print control object that contains an environment definition and one or more Medium Maps. Synonymous with *Form Definition*. See also *Medium Map*.

full arc. A complete circle or ellipse. See also *arc*.

fully described font. In the IPDS architecture, an LF1-type raster-font resource containing font metrics, descriptive information, and the raster representation of character shapes, for a specific graphic character set. A fully described font can be downloaded to a printer using the Load Font Control and Load Font commands. An LF1-type coded font or coded-font section is the combination of one fully described font and one font index. See also *font index*.

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*.

Global Identifier (GID). Any of the following:

- Code Page Global ID (CPGID)
- Graphic Character Global Identifier (GCGID)
- Graphic Character UCS Identifier (GCUID)
- Font Typeface Global Identifier (FGID)
- Graphic Character Set Global Identifier (GCSGID)
- Coded Graphic Character Set Global Identifier (CGCSGID)
- In MO:DCA, an encoded graphic character string that provides a reference name for a document element.
- Global Resource Identifier (GRID)
- Object identifier (OID)
- Coded Character Set Identifier (CCSID).

global resource identifier (GRID). An eight-byte identifier that identifies a coded font resource. A GRID contains the following fields in the order shown:

1. GCSGID of a minimum set of graphic characters required for presentation. It can be a character set that is associated with the code page, or with the font character set, or with both.
2. CPGID of the associated code page
3. FGID of the associated font character set
4. Font width in 1440ths of an inch.

glyph. 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*.

GOCA. See *Graphics Object Content Architecture*.

graphic character. A member of a set of symbols that represent data. Graphic characters can be letters, digits, punctuation marks, or other symbols. Synonymous with *glyph*. See also *character*.

Graphic Character Global Identifier (GCGID). An alphanumeric character string used to identify a specific graphic character. A GCGID can be from four-bytes to eight-bytes long.

graphic character identifier. 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 command set. In the IPDS architecture, a collection of commands used to present GOCA data in a page, page segment, or overlay.

graphics data. Data containing lines, arcs, markers, and other constructs that describe a picture.

graphics model space. A two-dimensional conceptual space in which a picture is constructed. All model transforms are completed before a picture is constructed in a graphics model space. Contrast with *graphics presentation space*. Synonymous with *model space*.

graphics object. An object that contains graphics data. See also *object*.

graphics object area. A rectangular area on a logical page into which a graphics presentation space window is mapped.

Graphics Object Content Architecture (GOCA). An architected collection of constructs used to interchange and present graphics data.

graphics presentation space. A two-dimensional conceptual space in which a picture is constructed. In this space graphics drawing orders are defined. The picture can then be mapped onto an output medium. All viewing transforms are completed before the picture is generated for presentation on an output medium. An example of a graphics presentation space is the abstract space containing graphics pictures defined in an IPDS Write Graphics Control command. Contrast with *graphics model space*.

graphics presentation space window. The portion of a graphics presentation space that can be mapped to a graphics object area on a logical page.

graphics primitive. A basic construct used by an output device to draw a picture. Examples of graphics primitives are arc, line, fillet, character string, and marker.

graphics processor. The processing capability required to interpret a GOCA object, that is, to present the picture represented by the object. It includes the environment interface, which interprets commands and instructions, and the drawing processor, which interprets the drawing orders.

graphics segment. A set of graphics drawing orders contained within a Begin Segment command. See also *segment*.

grayscale image. Images whose image data elements are represented by multiple bits and whose image data element values are mapped to more than one level of brightness through an image data element structure parameter or a look-up table.

GRID. See *global resource identifier*.

guard bars. The bars at both ends and the center of an EAN, JAN, or UPC symbol, that provide reference points for scanning.

H

HAID. See *Host-Assigned ID*.

height. In bar codes, the bar dimension perpendicular to the bar width. Synonymous with *bar height* and *bar length*.

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*.

highlighting. The emphasis of displayed or printed information. Examples are increased intensity of selected characters on a display screen and exception highlighting on an IPDS printer.

hollow font. A font design in which the graphic character shapes include only the outer edges of the strokes.

home state. An initial IPDS operating state. A printer returns to home state at the end of each page, and after downloading a font, overlay, or page segment.

horizontal bar code. A bar code pattern presenting the axis of the symbol in its length dimension parallel to the X_{bc} axis of the bar code presentation space. Synonymous with *picket fence bar code*.

horizontal font size.

1. A characteristic value, parallel to the character baseline, that represents the size of all graphic characters in a font. Synonymous with *font width*.
2. In a font character set, nominal horizontal font size is a font-designer defined value corresponding to the nominal character increment for a font character set. The value is generally the width of the space character and is defined differently for fonts with different spacing characteristics.
   - For fixed-pitch, uniform character increment fonts: the fixed character increment, which is also the space character increment
   - For PSM fonts: the width of the space character
   - For typographic, proportionally spaced fonts: one-third of the vertical font size, which is also the default size of the space character.

   The font designer can also define a minimum and a maximum horizontal font size to represent the limits of scaling.
3. In font referencing, the specified horizontal font size is the desired size of the font when the characters are presented. If this size is different from the nominal horizontal font size specified in a font character set, the character shapes and character metrics might need to be scaled prior to presentation.

horizontal scale factor.

1. In outline-font referencing, the specified horizontal adjustment of the Em square. The horizontal scale factor is specified in 1440ths of an inch. When the horizontal and vertical scale factors are different, anamorphic scaling occurs. See also *vertical scale factor*.
2. In FOCA, the numerator of a scaling ratio, determined by dividing the horizontal scale factor by the vertical font size. If the value specified is greater or less than the specified vertical font size, the graphic characters and their corresponding metric values are stretched or compressed in the horizontal direction relative to the vertical direction by the scaling ratio indicated.

host.

1. In the IPDS architecture, a computer that drives a printer.
2. In IOCA, the host is the controlling environment.

Host-Assigned ID (HAID). A two-byte ID assigned by the host to a font, page segment, or overlay. This ID is used when loading a resource and to identify a resident font or page segment.

Host-Assigned Resource ID. The combination of a Host-Assigned ID with a section identifier, or a font inline sequence, or both. The section identifier and font inline sequence values are ignored for both page segments and overlays. See also *section identifier* and *font inline sequence*.

HRI. See *human-readable interpretation*.

human-readable interpretation (HRI). The printed translation of bar code characters into equivalent Latin alphabetic characters, Arabic numeral decimal digits, and common special characters normally used for printed human communication.

hypermedia. Interlinked pieces of information consisting of a variety of data types such as text, graphics, image, audio, and video.

hypertext. Interlinked pieces of information consisting primarily of text.

I

i_c. See *current inline print coordinate*.

i_i. See *initial inline print coordinate*.

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. See also *Host-Assigned ID (HAID)*, *correlation ID*, *font control record*, and *overlay ID*.

IDE. See *image data element*.

IEEE. Institute of Electrical and Electronics Engineers.

I-direction.

1. The direction in which successive characters appear in a line of text.
2. In GOCA, the direction specified by the character angle attribute. Synonymous with *inline direction*.

IDP. See *image data parameter*.

I-extent. The X_p-extent when the I axis is parallel to the X_p axis or the 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 an 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 content. Image data and its associated image data parameters.

image coordinate system. An X,Y Cartesian coordinate system using only the fourth quadrant with positive values for the Y axis. The origin of an image coordinate system is its upper left hand corner. An X,Y coordinate specifies a presentation position that corresponds to one and only one image data element in the image content.

image data. Rectangular arrays of raster information that define an image.

image data element (IDE). A basic unit of image information. An image data element expresses the intensity of a signal at a corresponding image point. An image data element can use a look-up table to introduce a level of indirection into the expression of grayscale or color.

image data parameter (IDP). A parameter that describes characteristics of image data.

image distortion. Deformation of an image such that the original proportions of the image are changed and the original balance and symmetry of the image are lost.

image object. An object that contains image data. See also *object*.

image object area. A rectangular area on a logical page into which an image presentation space is mapped.

Image Object Content Architecture (IOCA). An architected collection of constructs used to interchange and present images.

image point. A discrete X,Y coordinate in the image presentation space. See also *addressable position*.

image presentation space (IPS). A two-dimensional conceptual space in which an image is generated.

image segment. Image content bracketed by Begin Segment and End Segment self-defining fields. See also *segment*.

IM image. A migration image object that is resolution dependent, bilevel, and cannot be compressed or scaled. Contrast with *IO image*.

IM-image command set. In the IPDS architecture, a collection of commands used to present IM-image data in a page, page segment, or overlay.

immediate mode. The mode in which segments are executed as they are received and then discarded. Contrast with *store mode*.

indexed object. An object in a MO:DCA document that is referenced by an Index Element structured field in a MO:DCA index. Examples of indexed objects are pages and page groups.

information density. The number of characters per inch (cpi) in a bar code symbology. In most cases, the range is three to ten cpi. See also *bar code density*, *character density*, and *density*.

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.

initial baseline print coordinate (b_i). The baseline coordinate of the first print position on a logical page. See also *initial inline print coordinate*.

initial inline print coordinate (i_i). The inline coordinate of the first print position on a logical page. See also *initial baseline print coordinate*.

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).

1. The direction in which successive characters appear in a line of text.
2. In GOCA, the direction specified by the character angle attribute. 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 gap. In bar codes, the space between two adjacent bar code characters in a discrete code, for example, the space between two characters in Code 39. Synonymous with *intercharacter space*. Contrast with *clear area*, *element*, and *space*.

intercharacter increment. Intercharacter adjustment applied in the positive I-direction from the current presentation position. See also *intercharacter adjustment*.

intercharacter space. In bar codes, the space between two adjacent bar code characters in a discrete code, for example, the space between two characters in Code 39. Synonymous with *intercharacter gap*. Contrast with *element* and *space*.

interleaved bar code. A bar code symbology in which characters are paired, using bars to represent the first character and spaces to represent the second. An example is Interleaved 2 of 5.

intermediate device. In the IPDS architecture, a device that operates on the data stream and is situated between a printer and a presentation services program in the host. Examples include devices that capture and cache resources and devices that spool the data stream.

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*.

International Organization for Standardization (ISO). An organization of national standards bodies from various countries established to promote development of standards to facilitate international exchange of goods and services, and develop cooperation in intellectual, scientific, technological, and economic activity.

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.

introducer. In GOCA, that part of the data stream passed from a controlling environment to a communication processor that indicates whether entities are to be processed in immediate mode or store mode. See also *immediate mode* and *store mode*.

IOCA. See *Image Object Content Architecture*.

IO image. An image object containing IOCA constructs. Contrast with *IM image*.

IO-image command set. In the IPDS architecture, a collection of commands used to present IOCA data in a page, page segment, or overlay.

IPDS. See *Intelligent Printer Data Stream*.

IPS. See *image presentation space*.

ISO. See *International Organization for Standardization*.

italics. A typeface with characters that slant upward to the right. In FOCA, italics is the common name for the defined inclined typeface posture attribute or parameter.

J

JAN. See *Japanese Article Numbering*.

Japanese Article Numbering (JAN). The bar code symbology used to code grocery items in Japan.

jog. To cause printed sheet s to be stacked in an output stacker offset from previously stacked sheets. Jogging is requested by using an IPDS Execute Order Anystate Alternate Offset Stacker command.

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.

kerning track. A straight-line graph that associates vertical font size with white space adjustment. The result of this association is used to scale fonts.

kerning track intercept. The X-intercept of a kerning track for a given vertical font size or white space adjustment value.

kerning track slope. The slope of a kerning track.

keyword. A two-part self-defining parameter consisting of a one-byte identifier and a one-byte value.

L

ladder bar code. A bar code pattern presenting the axis of the symbol in its length dimension parallel to the Y_{bc} axis of the bar code presentation space. Synonymous with *vertical bar code*.

LAN. See *local area network*.

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*.

LCID. See *Local Character Set Identifier*.

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*.

line attributes. Those attributes that pertain to straight and curved lines. Examples of line attributes are line type and line width.

line type. A line attribute that controls the appearance of a line. Examples of line types are dashed, dotted, and solid. Contrast with *line width*.

line width. A line attribute that controls the appearance of a line. Examples of line width are normal and thick. Contrast with *line type*.

link. A logical connection from a source document component to a target document component.

loaded-font command set. In the IPDS architecture, a collection of commands used to load font information into a printer and to deactivate font resources.

local area network (LAN). A data network located on a user's premises in which serial transmission is used for direct data communication among data stations.

Local Character Set Identifier (LCID). A local identifier used as a character, marker, or pattern set attribute.

local extent. In FD:OCA, the number of subpartitions within any given partition.

local identifier (LID). An identifier that is mapped by the environment to a named resource.

location. A site within a data stream. A location is specified in terms of an offset in the number of structured fields from the beginning of a data stream, or in the number of bytes from another location within the data stream.

logical page. A presentation space. One or more object areas 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 = 14400)

1 logical unit = 1/240 inch
   (unit base = 10 inches,
    units per unit base = 2400)
```

Synonymous with *L-unit*.

look-up table (LUT). A logical list of colors or intensities. The list has a name and can be referenced to select a color or intensity. See also *color table*.

lossless. A form of image transformation in which all of the data is retained. Contrast with *lossy*.

lossy. A form of image transformation in which some of the data is lost. Contrast with *lossless*.

lowercase. Pertaining to small letters as distinguished from capital letters. Examples of small letters are *a*, *b*, and *g*. Contrast with *uppercase*.

L-unit. A unit of linear measurement expressed with a unit base and units per unit-base value. For example, in MO:DCA and IPDS architectures, the following L-units are used:

```
1 L-unit = l/1440 inch
   (unit base = 10 inches,
    units per unit base = 14400)

1 L-unit = 1/240 inch
   (unit base = 10 inches,
    units per unit base = 2400)
```

Synonymous with *logical unit*.

LUT. See *look-up table*.

M

magnetic ink character recognition (MICR). Recognition of characters printed with ink that contains particles of a magnetic material.

mainframe interactive (MFI). Pertaining to systems in which nonprogrammable terminals are connected to a mainframe.

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*.

marker. A symbol with a recognizable appearance that is used to identify a particular location. An example of a marker is a symbol that is positioned by the center point of its cell.

marker attributes. The characteristics that control the appearance of a marker. Examples of marker attributes are size and color.

marker cell. A conceptual rectangular box that can include a marker symbol and the space surrounding that symbol.

marker precision. A method used to specify the degree of influence that marker attributes have on the appearance of a marker.

marker set. In GOCA, an attribute used to access a coded font.

marker symbol. A symbol that is used for a marker.

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*.

media destination. The destination to which sheets are sent as the last step in the print process. Some printers support several media destinations to allow options such as print job distribution to one or more specific destinations, collated copies without having to resend the document to the printer multiple times, and routing output to a specific destination for security reasons. Contrast with *media source*.

media source. The source from which sheets are obtained for printing. Some printers support several

media sources so that media with different characteristics (such as size, color, and type) can be selected when desired. Contrast with *media destination*.

medium. A two-dimensional conceptual space with a base coordinate system from which all other coordinate systems are either directly or indirectly derived. A medium is mapped onto a physical medium in a device-dependent manner. Synonymous with *medium presentation space*. See also *logical page*, *physical medium*, and *presentation space*.

Medium Map. A print control object in a Form Map that defines resource mappings and controls modifications to a form, page placement on a form, and form copy generation. See also *Form Map*.

medium presentation space. A two-dimensional conceptual space with a base coordinate system from which all other coordinate systems are either directly or indirectly derived. A medium presentation space is mapped onto a physical medium in a device-dependent manner. Synonymous with *medium*. See also *logical page*, *physical medium*, and *presentation space*.

MFI. See *mainframe interactive*.

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*.

mixing.

1. Combining foreground and background of one presentation space with foreground and background of another presentation space in areas where the presentation spaces intersect.
2. Combining foreground and background of multiple intersecting object data elements in the object presentation space.

mixing rule. A method for specifying the color attributes of the resulting foreground and background in areas where two presentation spaces intersect.

Mixed Object Document Content Architecture (MO:DCA). An architected, device-independent data stream for interchanging documents.

MO:DCA. See *Mixed Object Document Content Architecture*.

MO:DCA-L. MO:DCA Resource Interchange Set. A subset of MO:DCA that defines an interchange format for resource documents. Contrast with *MO:DCA-P IS/1* and *MO:DCA-P IS/2*.

MO:DCA-P. The subset of MO:DCA that defines presentation documents.

MO:DCA-P IS/1. MO:DCA Presentation Interchange Set 1. A subset of MO:DCA-P that defines an interchange format for presentation documents. See also *MO:DCA-P IS/2*. Contrast with *MO:DCA-L*.

MO:DCA-P IS/2. MO:DCA Presentation Interchange Set 2. A subset of MO:DCA-P that defines an interchange format for presentation documents and is a superset of MO:DCA-P IS/1. See also *MO:DCA-P IS/1*. Contrast with *MO:DCA-L*.

model space. A two-dimensional conceptual space in which a picture is constructed. All model transforms are completed before a picture is constructed in a graphics model space. Contrast with *graphics presentation space*. Synonymous with *graphics model space*.

model transform. A transform that is applied to drawing-order coordinates. Contrast with *viewing transform*.

module. In a bar code symbology, the nominal width of the smallest element of a bar or space. Actual bar code symbology bars and spaces can be a single module wide or some multiple of the module width. The multiple need not be an integer.

modulo-N check. A check in which an operand is divided by a modulus to generate a remainder that is retained and later used for checking. An example of an operand is the sum of a set of digits. See also *modulus*.

modulus. In a modulo check, the number by which an operand is divided. An example of an operand is the sum of a set of digits. See also *modulo-N check*.

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*.

move order. A drawing order that specifies or implies movement from the current position to a given position. See also *current position* and *given position*.

N

NACK. See *Negative Acknowledge Reply*.

name. A table heading for architecture syntax. The entries under this heading are short names that give a general indication of the contents of the construct.

named color. A color that is specified with a descriptive name. An example of a named color is "green".

navigation. The traversing of a document based on links between contextually related document components.

navigation link. A link type that specifies the linkage from a source document component to a contextually related target document component. Navigation links can be used to support applications such as hypertext and hypermedia.

Negative Acknowledge Reply (NACK). In the IPDS architecture, a reply from a printer to a host, indicating that an exception has occurred. Contrast with *Positive Acknowledge Reply*.

nested resource. A resource that is invoked within another resource using either an Include command or a local ID. See also *nesting resource*.

nesting coordinate space. A coordinate space that contains another coordinate space. Examples of coordinate spaces are medium, overlay, page, and object area.

nesting resource. A resource that invokes nested resources. See also *nested resource*.

neutral white. A color attribute that gives a device-dependent default color, typically white on a screen and black on a printer.

nonprocess runout (NPRO). An operation that moves sheets of physical media through the printer without printing on them. This operation is used to stack the last printed sheet.

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.

NPRO. See *nonprocess runout*.

N-up. The partitioning of a side of a sheet into a fixed number of equal size partitions. For example, 4-up divides each side of a sheet into four equal partitions.

O

object.

1. A collection of structured fields. The first structured field provides a begin-object function, and the last structured field provides an end-object function. The object can contain one or more other structured fields whose content consists of one or more data elements of a particular data type. An object can be assigned a name, which can be used to reference the object. Examples of objects are text, font, graphics, image, and formatted data objects.
2. Something that a user works with to perform a task.

object area. A rectangular area in a presentation space into which a data object is mapped. The presentation space can be for a page or an overlay. Examples are a graphics object area, an image object area, and a bar code object area.

object data. A collection of related data elements bundled together. Examples of object data include graphic characters, image data elements, and drawing orders.

object identifier (OID). A notation that assigns a globally unambiguous name to an object or a document component. The notation is defined in international standard ISO/IEC 8824(E).

OCR-A. See *Optical Character Recognition-A*.

OCR-B. See *Optical Character Recognition-B*.

offline. A device state in which the device is not under the direct control of a host. Contrast with *online*.

offset. A table heading for architecture syntax. The entries under this heading indicate the numeric displacement into a construct. The offset is measured in bytes and starts with byte zero. Individual bits can be expressed as displacements within bytes.

OID. See *object identifier*.

online. A device state in which the device is under the direct control of a host. Contrast with *offline*.

opacity. In bar codes, the optical property of a substrate material that minimizes showing through from the back side or the next sheet.

Optical Character Recognition-A (OCR-A). A font containing the character set in ANSI standard X3.17-1981, that contains characters that are both human-readable and machine-readable.

Optical Character Recognition-B (OCR-B). A font containing the character set in ANSI standard X3.49-1975, that contains characters that are both human-readable and machine-readable.

order.

1. In GOCA, a graphics construct that the controlling environment builds to instruct a drawing processor about what to draw and how to draw it. The order can specify, for example, that a graphics primitive be drawn, a change to drawing attributes or drawing controls be effected, or a segment be called. One or more graphics primitives can be used to draw a picture. Orders can be included in a structured field. Synonymous with *drawing order*.

2. In the IPDS architecture, a construct within an execute-order command.
3. In IOCA, a functional operation that is performed on the image content.

ordered page. In the IPDS architecture, a logical page that does not contain any page segments or overlays, and in which all text data and all image, graphics, and bar code objects are ordered. The order of the data objects is such that physical pel locations on the physical medium are accessed by the printer in a sequential left-to-right and top-to-bottom manner, where these directions are relative to the top edge of the physical medium. Once a physical pel location has been accessed by the printer, the page data does not require the printer to access that same physical pel location again.

orientation. The angular distance a presentation space or object area is rotated in a specified coordinate system, expressed in degrees and minutes. For example, the orientation of printing on a physical medium, relative to the X_m axis of the X_m,Y_m coordinate system. 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.

outline font. A shape technology in which the graphic character shapes are represented in digital form by a series of mathematical expressions that define the outer edges of the strokes. The resultant graphic character shapes can be either solid or hollow.

overhead. In bar code symbologies, the fixed number of characters required for starting, stopping, and checking a bar code symbol.

overlay.

1. A resource object that contains presentation data such as, text, image, graphics, and bar code data. Overlays define their own environment and are often used as pre-defined pages or electronic forms. Overlays are classified according to how they are presented with other presentation data: a medium overlay is positioned at the origin of the medium presentation space before any pages are presented, and a page overlay is positioned at a specified point in a page's logical page. A Page Modification Control (PMC) overlay is a special type of page overlay used in MO:DCA environments.
2. The final representation of such an object on a physical medium. Contrast with *page segment*.

overlay command set. In the IPDS architecture, a collection of commands used to load, deactivate, and include overlays.

overlay ID. A one-byte ID assigned by a host to an overlay. Overlay IDs are used in IPDS Begin Overlay, Deactivate Overlay, Include Overlay, and Load Copy Control commands.

overlay state. An operating state that allows overlay data to be downloaded to a product. For example, a printer enters overlay state from home state when the printer receives an IPDS Begin Overlay command.

overpaint. A mixing rule in which the intersection of part of a new presentation space P_{new} with an existing presentation space $P_{existing}$ keeps the color attribute of P_{new}. This is also referred to as "opaque" mixing. See also *mixing rule*. Contrast with *underpaint*.

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 blanks, 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 presentation data such as text, image, graphics, and bar code data.
2. The final representation of a page object on a physical medium.

page counter. Bytes in an IPDS Acknowledge Reply that specify the number of pages that have passed a particular point in a logical paper path.

page group. A named group of sequential pages. A page group is delimited by a Begin Named Page Group structured field and an End Named Page Group structured field. A page group can contain nested page groups. All pages in the page group inherit the attributes and processing characteristics that are assigned to the page group.

page segment.

1. In the 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 MO:DCA, a resource object that can contain any mixture of bar code objects, graphics objects, and IOCA image objects. A page segment does not contain an active environment group. The environment for a page segment is defined by the active environment group of the including page or overlay.
3. The final representation of such an object on a physical medium. Contrast with *overlay*.

page-segment command set. In the IPDS architecture, a collection of commands used to load, deactivate, and include page segments.

page-segment state. An operating state that makes page-segment data available to a product. For example, a printer enters page-segment state from home state when it receives an IPDS Begin Page Segment command.

page state. In the IPDS architecture, an operating state that makes page data available to a product. For example, a printer enters page state from home state when it receives an IPDS Begin Page command.

parameter.

1. A variable that is given a constant value for a specified application.
2. A variable used in conjunction with a command to affect its result.

partition.

1. Dividing the medium presentation space into a specified number of equal-sized areas in a manner determined by the current physical media.
2. In FD:OCA, a conceptual subdivision of a string of data fields. A partition can be further divided into subpartitions. See also *dimension*.

partitioning.

1. A method used to place parts of a control into two or more segments or structured fields. Partitioning can cause difficulties for a receiver if one of the segments or structured fields is not received or is received out of order.
2. In FD:OCA, a conceptual division of a string of data fields into substrings. Each substring is called a partition. See also *partition*.

pattern. An array of symbols used to fill an area.

pattern attributes. The characteristics that specify the appearance of a pattern.

pattern set. An attribute in GOCA used to access a symbol set or coded font.

pattern symbol. The geometric construct that is used repetitively to generate a pattern. Examples of pattern symbols are dots, squares, and triangles.

PCS. See *Print Contrast Signal*.

pel. The smallest printable or displayable unit on a physical medium. In computer graphics, the smallest element of a physical medium that can be independently assigned color and intensity. Pels per inch is often used as a measurement of presentation granularity. Synonymous with *picture element* and *pixel*.

physical medium. A physical entity on which information is presented. Examples of a physical medium are a sheet of paper, a roll of paper, an envelope, and a display screen. See also *medium presentation space* and *sheet*.

physical printable area. A bounded area defined on a side of a sheet within which printing can take place. The physical printable area is an attribute of sheet size and printer capabilities, and cannot be altered by the host. The physical printable area is mapped to the medium presentation space, and is used in user printable area and valid printable area calculations. Contrast with *user printable area* and *valid printable area*.

pick. A match between the pick window and a graphics primitive during correlation.

pick identifier. An identifier put at a particular position in a drawing order sequence so that the position of a pick in the picture chain can be easily recognized.

pick window. A region of a graphics presentation space that is used for correlation. A pick window has specified characteristics. Examples of pick window characteristics are position in a graphics presentation space and size. See also *correlation*.

pickable segment. A segment whose graphics primitives are eligible to be picked during correlation. See also *correlation*.

picket fence bar code. A bar code pattern presenting the axis of the symbol in its length dimension parallel to the X_{bc} axis of the bar code presentation space. Synonymous with *horizontal bar code*.

picture chain. A string of segments that defines a picture. Synonymous with *segment chain*.

picture element. 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. Picture elements per inch is often used as a measurement of presentation granularity. Synonymous with *pel* and *pixel*.

pixel. 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. Picture elements per inch is often used as a measurement of presentation granularity. Synonymous with *pel* and *picture element*.

plane. In FD:OCA, a two-dimensional subarray consisting of all elements that have an identical position within a given dimension of a regular three-dimensional array.

point. (1) A unit of measure used mainly for measuring typographical material. There are seventy-two points to an inch. (2) In GOCA, a parameter that specifies the position within the drawing order coordinate space. See also *drawing order coordinate space*.

polyline. A sequence of connected lines.

pop. A method used to retrieve a value from a segment call stack. Contrast with *push*.

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*.

Positive Acknowledge Reply (ACK). In the IPDS architecture, a reply to an IPDS command that has its ARQ flag on and in which no exception is reported. Contrast with *Negative Acknowledge Reply*.

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. See also *semantics* and *syntax*.

presentation device. A device that produces character shapes, graphics pictures, images, or bar code symbols on a physical medium. Examples of 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 *graphics presentation space*, *image presentation space*, *logical page*, *medium presentation space*, and *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 contrast. A measurement of the ratio of the reflectivities between the bars and spaces of a bar code symbol, commonly expressed as a percent. Synonymous with *Print Contrast Signal*.

Print Contrast Signal (PCS). A measurement of the ratio of the reflectivities between the bars and spaces of a bar code symbol, commonly expressed as a percent. Synonymous with *print contrast*.

print control object. A resource object that contains layout, finishing, and resource mapping information used to present a document on physical media. Examples of print control objects are Form Maps and Medium Maps.

print direction. In FOCA, the direction in which successive characters appear in a line of text.

print quality. In bar codes, the measure of compliance of a bar code symbol to the requirements of dimensional tolerance, edge roughness, spots, voids, reflectivity, PCS, and quiet zones defined within a bar code symbology.

printing baseline. A conceptual line with respect to which successive characters are aligned. See also *character baseline*. Synonymous with *baseline* and *sequential baseline*.

print unit. In the IPDS architecture, a group of pages bounded by XOH-DGB commands and subject to the group operation “keep group together as a print unit”. A print unit is commonly referred to as a “print job”.

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*.

process element. In MO:DCA, a document component that is defined by a structured field and that facilitates a form of document processing that does not affect the presentation of the document. Examples of process elements are Tag Logical Elements (TLEs) that specify document attributes and Link Logical Elements (LLEs) that specify linkages between document components.

prolog. The first portion of a segment's data. Prologs are optional. They contain attribute settings and drawing controls. Synonymous with *segment prolog*.

propagation. A method used to retain a segment's properties through other segments that it calls.

proper subset. A set whose members are also members of a larger set.

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.

Proportional Spacing Machine font (PSM font). A font originating with the electric typewriter and having character increment values that are integer multiples of the narrowest character width.

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*.

PSM font. See *Proportional Spacing Machine font*.

PTOCA. See *Presentation Text Object Content Architecture*.

push. A method used to store a current value on a segment call stack. Contrast with *pop*.

pushdown list. A list that is constructed and maintained so that the next item to be retrieved and removed is the most recently stored item still in the list. This is sometimes called last-in-first-out (LIFO). Synonymous with *stack*. See also *segment call stack*.

Q

quiet zone. A clear space that contains no machine-readable marks preceding the start character of a bar code symbol or following the stop character. Synonymous with *clear area*. Contrast with *intercharacter gap* and *space*.

R

range. A table heading for architecture syntax. The entries under this heading give numeric ranges applicable to a construct. The ranges can be expressed in binary, decimal, or hexadecimal. The range can consist of a single value.

raster pattern. A rectangular array of pels arranged in rows called scan lines.

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.

reader. In bar code systems, the scanner or combination of scanner and decoder. See also *decoder* and *scanner*.

read rate. In bar codes, the ratio of the number of successful reads on the first attempt to the total number of attempts made to obtain a successful read. Synonymous with *first read rate*.

recording algorithm. An algorithm that determines the relationship between the physical location and logical location of image points in image data.

redaction. The process of applying an opaque mask over a page so that a selected portion of the page is visible. Since this function is typically used to prevent unauthorized viewing of data, an associated security level is also provided.

reflectance. In bar codes, the ratio of the amount of light of a specified wavelength or series of wavelengths reflected from a test surface to the amount of light reflected from a barium oxide or magnesium oxide standard under similar illumination conditions.

regular array. In FD:OCA, an array in which all partitions of any dimension have the same number of subpartitions. The individual elements of a regular array might or might not have identical format and length. See also *array*.

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*.

resident resource. In the IPDS architecture, a resource in a printer or in a resource-caching intermediate device. A resident resource can be installed manually or can be captured by the device if it is intended for public use. A resident resource can not be removed by a presentation services program. A resident resource is referenced by a global ID that is valid for the duration of the resource's presence in the device. Contrast with *downloaded resource*.

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.

resolution correction. A method used to present an image on a printer without changing the physical size or proportions of the image when the resolutions of the printer and the image are different.

resolution-correction ratio. The ratio of a printer's physical resolution to an image presentation space's resolution.

resolution modification. A method used to write an image on an image presentation space without changing the physical size of the image when the resolutions of the presentation space and the image are different.

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 MO:DCA, resource objects can be contained within a resource group. Examples of resources are fonts, overlays, and page segments. See also *downloaded resource* and *resident resource*.

resource caching. In the IPDS architecture, a function in a printer or intermediate device whereby downloaded resources are captured and made resident in the printer or intermediate device.

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.

return address. The address of the order following a Call Segment order, that is pushed onto the segment call stack at call time. This enables a return from the called segment so that processing can resume with that order.

RM4SCC. See *Royal Mail 4 State Customer Code*.

Roman. Relating to a type style with upright letters.

root segment. A segment in the picture chain that is not called by any other segment. If a single segment that is not in a chain is drawn, it is treated as a root segment for the duration of the drawing process.

rotating. In computer graphics, turning all or part of a picture about an axis perpendicular to the presentation space.

rotation. The orientation of a presentation space with respect to the coordinate system of a containing presentation space. Rotation is measured in degrees in a clockwise direction. Zero-degree rotation exists when the angle between a presentation space's positive X axis and the containing presentation space's positive X axis is zero degrees. Contrast with *character rotation*.

row. A subarray that consists of all elements that have an identical position within the high dimension of a regular two-dimensional array.

Royal Mail 4 State Customer Code (RM4SCC). A two-dimensional bar code symbology developed by the United Kingdom's Royal Mail postal service for use in automated mail-sorting processes.

rule. A solid line of any line width.

S

SAA. See *Systems Application Architecture*.

SAA environments. Those environments in which IBM intends to provide full implementation of applicable SAA architectural elements. See also *interoperability*.

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 twos-complement binary notation with a B'1' in the sign-bit position.

scaling. Making all or part of a picture smaller or larger by multiplying the coordinate values of the picture by a constant amount. If the same multiplier is applied along both dimensions, the scaling is uniform, and the proportions of the picture are unaffected. Otherwise, the scaling is anamorphic, and the proportions of the picture are changed. See also *anamorphic scaling*.

scaling ratio.

1. The ratio of an image-object-area size to its image-presentation-space size.
2. In FOCA, the ratio of horizontal to vertical scaling of the graphic characters. See also *horizontal scale factor*.

scan line. A series of picture elements. Scan lines in raster patterns form images. See also *picture element* and *raster pattern*.

scanner. In bar codes, an electronic device that converts optical information into electrical signals. See also *reader*.

scrolling. A method used to move a displayed image vertically or horizontally so that new data appears at one edge as old data disappears at the opposite edge. Data disappears at the edge toward which an image is moved and appears at the edge away from which the data is moved.

SDA. See *special data area*.

section. A portion of a double-byte code page that consists of 256 consecutive entries. The first byte of a two-byte code point is the section identifier. A code-page section is also called a code-page ward in some environments. See also *code page* and *code point*.

section identifier. A value that identifies a section. Synonymous with *section number*.

section number. A value that identifies a section. Synonymous with *section identifier*.

secure overlay. An overlay that can be printed anywhere within the physical printable area. A secure overlay is not affected by an IPDS Define User Area command.

segment.

1. In GOCA, a set of graphics drawing orders contained within a Begin Segment command. See also *graphics segment*.
2. In IOCA, image content bracketed by Begin Segment and End Segment self-defining fields. See also *image segment*.

segment call stack. A pushdown list for storing specific current values, either when an attribute or drawing control is pushed onto the stack or when another segment is called.

segment chain. A string of segments that defines a picture. Synonymous with *picture chain*.

segment exception condition. An architecture-provided classification of the errors that can occur in a segment. Segment exception conditions are raised when a segment error is detected. Examples of segment errors are segment format, parameter content, and sequence errors.

segment offset. A position within a segment, measured in bytes from the beginning of the segment. The beginning of a segment is always at offset zero.

segment prolog. The first portion of a segment's data. Prologs are optional. They contain attribute settings and drawing controls. Synonymous with *prolog*.

segment properties. The segment characteristics used by a drawing process. Examples of segment properties are segment name, segment length, chained, dynamic, highlighted, pickable, propagated, and visible.

segment transform. A model transform that is applied to a whole segment.

self-checking. In bar codes, using a checking algorithm that can be applied to each character independently to guard against undetected errors.

semantics. The meaning of the parameters of a construct. See also *pragmatics* and *syntax*.

sequential baseline. A conceptual line with respect to which successive characters are aligned. See also *character baseline*. Synonymous with *baseline* and *printing 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*.

session. In the IPDS architecture, the period of time during which a presentation services program has sole control of a printer and has established two-way communication with the printer.

shade. Variation of a color produced by mixing it with black.

shape compression. A method used to compress digitally encoded character shapes using a specified algorithm.

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 attribute s.

sheet. A division of the physical medium; multiple sheets can exist on a physical medium. For example, a roll of paper might be divided by a printer into rectangular pieces of paper, each representing a sheet. Envelopes are an example of a physical medium that comprises only one sheet. The IPDS architecture defines four types of sheets: cut-sheets, continuous forms, envelopes, and computer output on microfilm. Each type of sheet has a top edge. A sheet has two sides, a front side and a back side. Synonymous with *form*.

show-through. In bar codes, the generally undesirable property of a substrate that permits underlying markings to be seen.

side. A physical surface of a sheet. A sheet has a front side and a back side. See also *sheet*.

simplex printing. A method used to print data on one side of a sheet; the other side is left blank. Contrast with *duplex printing*.

single-byte character set (SBCS). A character set that can contain up to 256 characters.

single-byte coded font. A coded font in which the code points are one byte long.

slice. In FD:OCA, a subarray that consists of all elements that have an identical position within any given dimension of a regular *n*-dimensional array.

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.

space. In bar codes, the lighter element of a printed bar code symbol, usually formed by the background between bars. See also *element*. Contrast with *bar*, *clear area*, *intercharacter gap*, and *quiet zone*.

space width. In bar codes, the thickness of a bar code symbol space measured from the edge closest to the symbol start character to the trailing edge of the same space.

spanning. In the IPDS architecture, a method in which one command is used to start a sequence of constructs. Subsequent commands continue and terminate that sequence. See also *control sequence chaining*.

special data area (SDA). The data area in an IPDS Acknowledge Reply that contains data requested by the host or generated by a printer as a result of an exception.

spot. In bar codes, the undesirable presence of ink or dirt in a bar code symbol space.

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*.

stack. A list that is constructed and maintained so that the next item to be retrieved and removed is the most recently stored item still in the list. This is sometimes called last-in-first-out (LIFO). Synonymous with *pushdown list*. See also *segment call stack*.

standard action. The architecture-defined action to be taken on detecting an exception condition, when the environment specifies that processing should continue.

start-stop character or pattern. In bar codes, a special bar code character that provides the scanner with start and stop reading instructions as well as a scanning direction indicator. The start character is normally at the left end and the stop character at the right end of a horizontally oriented symbol.

store mode. A mode in which segments are stored for later execution. Contrast with *immediate mode*.

stroke. A straight or curved line used to create the shape of a letter.

structured field. A self-identifying, variable-length, bounded record, which can have a content portion that provides control information, data, or both. See also *document element*.

structured field introducer. In MO:DCA, the header component of a structured field that provides information that is common for all structured fields. Examples of information that is common for all structured fields are length, function type, and category type. Examples of structured field function types are begin, end, data, and descriptor. Examples of structured field category types are presentation text, image, graphics, and page.

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*.

substrate. In bar codes, the surface on which a bar code symbol is printed.

suppression. A method used to prevent presentation of specified data. Examples of suppression are the processing of text data without placing characters on a physical medium and the electronic equivalent of the "spot carbon," that prevents selected data from being presented on certain copies of a presentation space or a physical medium.

symbol.

1. A visual representation of something by reason of relationship, association, or convention.
2. In GOCA, the subpicture referenced as a character definition within a font character set and used as a character, marker, or fill pattern. A bitmap can also be referenced as a symbol for use as a fill pattern. See also *bar code symbol*.

symbol length. In bar codes, the distance between the outside edges of the quiet zones of a bar code symbol.

symbol set. A coded font that is usually simpler in structure than a fully described font. Symbol sets are used where typographic quality is not required. Examples of devices that might not provide typographic quality are dot-matrix printers and displays. See also *character set*, *marker set*, and *pattern set*.

symbology. A bar code language. Bar code symbologies are defined and controlled by various industry groups and standards organizations. Bar code symbologies are described in public domain bar code specification documents. Synonymous with *bar code symbology*. See also *Canadian Grocery Product Code (CGPC)*, *European Article Numbering (EAN)*, *Japanese Article Numbering (JAN)*, and *Universal Product Code (UPC)*.

synchronous exception. In the IPDS architecture, a data-stream or resource-storage exception that must be reported to the host before a printer can return a Positive Acknowledge Reply or can increment the received-page counter for a page containing the exception. Synchronous exceptions are those with action code X'01', X'0C', or X'1F'. See also *data-stream exception*. Contrast with *asynchronous exception*.

syntax. The rules governing the structure of a construct. See also *pragmatics* and *semantics*.

Systems Application Architecture (SAA). A set of IBM software interfaces, conventions, and protocols that provide a framework for designing and developing applications that are consistent across systems.

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

tag. In FD:OCA, a special attribute triplet that can be attached to attribute triplets to provide them with additional information. In DRDA for example, an FD:OCA Metadata Definition triplet can express that a particular character field is actually a timestamp.

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. An example of text is the data sent in an IPDS Write Text command.

text command set. In the IPDS architecture, a collection of commands used to present PTOCA text data in a page, page segment, or overlay.

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.

throughscore. A line parallel to the baseline and placed through the character.

tint. Variation of a color produced by mixing it with white.

toned. Containing marking agents such as toner or ink. Contrast with *untoned*.

transform. A modification of one or more characteristics of a picture. Examples of picture characteristics that can be transformed are position, orientation, and size. See also *model transform*, *segment transform*, and *viewing transform*.

transform matrix. A matrix that is applied to a set of coordinates to produce a transform.

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 viewing window or presentation space. See also *viewing window*. Synonymous with *clipping*.

triplet. A three-part self-defining variable-length parameter consisting of a length byte, an identifier byte, and one or more parameter-value bytes.

triplet identifier. A one-byte type identifier for a triplet.

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. A method used to simulate the effect of physically turning a sheet around the X_m axis.

twip. A unit of measure equal to 1/20 of a point. There are 1440 twips in one inch.

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, Helvetica Bold Condensed 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.

underpaint. A mixing rule in which the intersection of part of a new presentation space P_{new} with part of an existing presentation space $P_{existing}$ keeps the color

attribute of $P_{existing}$. This is also referred to as "transparent" or "leave alone" mixing. See also *mixing rule*. Contrast with *overpaint*.

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+0024. Unicode provides for three encoding forms (UTF-8, UTF-16, and UTF-32), described as follows:

UTF-8 A byte-oriented form that is designed for ease of use in traditional ASCII environments. Each UTF-8 code point contains from one to four bytes. All Unicode code points can be encoded in UTF-8 and all 7-bit ASCII characters can be encoded in one byte.

UTF-16 The default Unicode encoding. A fixed, two-byte Unicode encoding form that can contain surrogates and identifies the byte order of each UTF-16 code point via a Byte Order Mark in the first 2 bytes of the data.

UTF-16BE UTF-16 that uses big endian byte order; this is the byte order for all multi-byte data within AFP data streams. The Byte Order Mark is not necessary when the data is externally identified as UTF-16BE (or UTF-16LE).

UTF-16LE UTF-16 that uses little endian byte order.

UTF-32 A fixed, four-byte Unicode encoding form in which each UTF-32 code point is precisely identical to the Unicode code point.

UTF-32BE UTF-32 serialized as bytes in most significant byte first order (big endian). UTF-32BE is structurally the same as UCS-4.

UTF-32LE UTF-32 serialized as bytes in least significant byte first order (little endian).

surrogates Pairs of Unicode code points that allow for the encoding of as many as 1 million additional characters without any use of escape codes.

Uniform Symbol Specification (USS). A series of bar code symbology specifications published by AIM; currently included are USS-Interleaved 2 of 5, USS-39, USS-93, USS-Codabar, and USS-128.

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.

Universal Product Code (UPC). A standard bar code symbology, commonly used to mark the price of items in stores, that can be read and interpreted by a computer.

untoned. Unmarked portion of a physical medium. Contrast with *toned*.

UPA. See *user printable area*.

UPC. See *Universal Product Code*.

uppercase. Pertaining to capital letters. Examples of capital letters are *A*, *B*, and *C*. Contrast with *lowercase*.

upstream data. IPDS commands that exist in a logical path from a specific point in a printer back to, but not including, host presentation services.

usable area. An area on a physical medium that can be used to present data. See also *viewport*.

user printable area (UPA). The portion of the physical printable area to which user-generated data is restricted. See also *logical page*, *physical printable area*, and *valid printable area*.

USS. See *Uniform Symbol Specification*.

V

valid printable area (VPA). The intersection of a logical page with the area of the medium presentation space in which printing is allowed. If the logical page is a secure overlay, the area in which printing is allowed is the physical printable area. If the logical page is not a secure overlay and if a user printable area

is defined, the area in which printing is allowed is the intersection of the physical printable area with the user printable area. If a user printable area is not defined, the area in which printing is allowed is the physical printable area. See also *logical page*, *physical printable area*, *secure overlay*, and *user printable area*.

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 character 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*.

verifier. In bar code systems, a device that measures the bars, spaces, quiet zones, and optical characteristics of a bar code symbol to determine if the symbol meets the requirements of a bar code symbology, specification, or standard.

vertical bar code. A bar code pattern that presents the axis of the symbol in its length dimension parallel to the Y_{bc} axis of the bar code presentation space. Synonymous with *ladder bar code*.

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.

vertical scale factor. In outline-font referencing, the specified vertical adjustment of the Em square. The vertical scale factor is specified in 1440ths of an inch. When the horizontal and vertical scale factors are different, anamorphic scaling occurs. See also *horizontal scale factor*.

viewing transform. A transform that is applied to model-space coordinates. Contrast with *model transform*.

viewing window. That part of a model space that is transformed, clipped, and moved into a graphics presentation space.

viewport. The portion of a usable area that is mapped to the graphics presentation space window. See also *graphics model space* and *graphics presentation space*.

visibility. The property of a segment that declares whether the part of a picture defined by the segment is to be displayed or not displayed during the drawing process.

void. In bar codes, the undesirable absence of ink in a bar code symbol bar element.

VPA. See *valid printable area*.

W

ward. A deprecated term for section.

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 the 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*.

window. A predefined part of a graphics presentation space. See also *graphics presentation space window* and *pick window*.

writing mode. An identified mode for the setting of text in a writing system, usually corresponding to a

nominal escapement direction of the graphic characters in that mode; for example, left-to-right, right-to-left, top-to-bottom.

X

X_{bc} extent. The size of a bar code presentation space in the X_{bc} dimension. See also *bar code presentation space*.

X_{bc},Y_{bc} coordinate system. The bar code presentation space coordinate system.

X-dimension. In bar codes, the nominal dimension of the narrow bars and spaces in a bar code symbol.

X_g,Y_g coordinate system. In the IPDS architecture, the graphics presentation space coordinate system.

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_{io},Y_{io} coordinate system. The IO-image presentation space coordinate system.

X_m,Y_m coordinate system. (1) In the IPDS architecture, the medium presentation space coordinate system. (2) In MO:DCA, the medium coordinate system.

X_{oa},Y_{oa} coordinate system. The object area coordinate system.

X_{ol},Y_{ol} coordinate system. The overlay coordinate system.

X_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. The X_p,Y_p coordinate system origin is specified by an IPDS Logical Page Position command. See also *logical page*, *medium presentation space*, and *presentation space*.

X_{pg},Y_{pg} coordinate system. The coordinate system of a page presentation space. This coordinate system describes the size, position, and orientation of a page presentation space. Orientation of an X_{pg},Y_{pg} coordinate system is relative to an environment specified coordinate system, for example, an X_m,Y_m coordinate system.

Y

Y_{bc} extent. The size of a bar code presentation space in the Y_{bc} dimension. See also *bar code presentation space*.

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

Numerics

3-of-9 Code 109

A

algorithm, L-unit range conversion 20
architectures
  BCOCA 17
  IPDS 113
  MO:DCA 111
Australia Post Bar Code 28, 30, 51, 56, 57, 58, 93
Automatic Identification (AutoID) 9

B

bar code
  AutoID 9
  color values 55
  definition 9
  elements of a system 9
  in IPDS 113
  in MO:DCA-P documents 111
  information density 14
  modifiers 30
  presentation space 19
  presenting 9
  retrieving 9
  symbol data definition 63
  symbol descriptor definition 26
  symbol generation process 14
  symbologies 10
  symbology specification references 109
  types 28
bar code commands
  in IPDS 113
Bar Code Data Descriptor structured field, MO:DCA 112
Bar Code Data structured field, MO:DCA 112
Bar Code Object Content Architecture
  compliance 107
  definition 17
  general concepts 17
  generator rules 107
  measurements 19
  object processor 17
  presentation space 19
  receiver rules 107
  symbol orientation 22
  symbol placement 21
  symbol size 23
Bar Code Symbol Data (BSA) data structure
  Code 39 65
  definition 25
  flags 64
  format 63
  HRI 64
  origin, x 68
  origin, y 68
  position 64
  variable data 69
Bar Code Symbol Descriptor (BSD) data structure
  definition 26
  element height 57
  element height patterns 57
  format 26
  module width 56
bar code symbol suppression flag 65
BCD1 subset 25
BCOCA
  See Bar Code Object Content Architecture
BDA 112
BDD 112
binary 14
BSA 63
BSD 26

C

check digit calculation methods 59
check-digit 10, 18
Codabar 28, 30, 38, 93, 95, 109
Code 128 9, 28, 30, 39, 58, 93, 95, 100, 109
Code 39 9, 14, 28, 30, 31, 65, 93, 94, 110
Code 93 28, 30, 53, 58, 93, 96
color values 55
compliance with BCOCA 107
compliance with MO:DCA-P IS/2 111
coordinate systems
  orthogonal 19
  Xbc, Ybc coordinate system 19

D

Data Matrix 28, 30, 47, 54, 57, 58, 93, 95
Data Matrix special-function parameters 70
data structures
  BSA 25
  BSD 26
  definition 25, 26
  diagram, syntax iv
  field values calculation 20
  measurements 20
data types
  definition iv
data-check exceptions 105
decoder 9, 16
Delivery Point bar code 43
Dutch KIX bar code 44, 95

E

EAN 9, 28, 29, 30, 36, 41, 42, 54, 57, 58, 64, 93, 94, 109
element height patterns, BSD 57
element height, BSD 57
encoding
  module width technique 14
  NRZ technique 14
environment, IPDS 113
exception conditions
  bar code object processor 18

exception conditions *(continued)*
data-check 105
EC-0300 28, 101
EC-0400 54, 101
EC-0500 55, 101
EC-0505 27, 101
EC-0600 56, 101
EC-0605 27, 101
EC-0700 57, 102
EC-0705 27, 102
EC-0800 57, 102
EC-0900 58, 102
EC-0A00 68, 102
EC-0B00 30, 102
EC-0C00 69, 102
EC-0E00 32, 33, 102
EC-0F00 71, 72, 102
EC-0F01 72, 78, 103
EC-0F02 73, 78, 103
EC-0F03 73, 78, 103
EC-0F04 73, 79, 103
EC-0F05 78, 103
EC-0F06 83, 103
EC-0F07 83, 103
EC-0F08 83, 103
EC-0F09 84, 103
EC-0F0A 73, 74, 104
EC-0F0B 73, 104
EC-0F0C 85, 104
EC-0F0D 85, 86, 104
EC-1000 64, 105
EC-1100 21, 28, 105, 114
EC-2100 50, 69, 85, 105
specification-check 101
standard actions 101
Extended Code 39 31
Extended Code 93 53
extent 19, 27

F

first read rate 16
four-level code 14

G

generator rules 107

H

height multiplier 57
HRI suppression flag 64
HRI, local ID 54
human-readable interpretation (HRI) 18

I

Industrial 2-of-5 28, 30, 37, 93, 94, 109
information density 14
Intelligent Printer Data Stream
additional related commands 114
bar code command set 113
bar code object area 17
environment 113
intercharacter gap 18
Interleaved 2-of-5 9, 14, 28, 30, 37, 93, 94, 109
IPDS
See Intelligent Printer Data Stream

J

Japan Postal Bar Code 28, 30, 45, 54, 57, 58, 93, 95

K

KIX bar code 44, 95

L

ladder orientation 22
LID 54
linear measurements 19
linear symbologies 10
local ID 54
logical units (L-units)
BSD, x direction 27
BSD, y direction 27
definition 19
field values calculation 20
fields 20
presentation space 19

M

Matrix 2-of-5 28, 30, 37, 93, 94, 109
MaxiCode 28, 30, 48, 54, 56, 57, 58, 93, 95
MaxiCode special-function parameters 75
measurement base 19, 27
measurements 19
message 10
Mixed Object Document Content Architecture
bar codes 111
compliance 111
environment 111
object area 17
structured fields 112
MO:DCA
See Mixed Object Document Content Architecture
modifiers 30
module width
BSD 56
encoding 14
module width encoding technique 14
MSI 28, 29, 30, 32, 58, 93, 94, 102, 110

N

non-return-to-zero encoding (NRZ) 14
notation conventions vi
notices 117
NRZ encoding technique 14

O

object processor
compliance rules 107
definition 17
orientation
ladder 22

orientation *(continued)*
picket fence 22

P

pattern, bar and space 14, 18, 57
PDF417 28, 29, 30, 49, 54, 58, 93, 96
PDF417 special-function parameters 81
performance measurement 16
physical media 9, 15
picket fence orientation 22
PLANET bar code 29, 43, 110
position HRI flags 64
POSTNET 28, 29, 30, 43, 54, 56, 57, 58, 93, 95, 109
presentation space
bar code object processor 18
coordinate system 19
definition 19
measurements 19
size 19
presenting bar code data 9
print quality 15
printers 15
printing 9
processor, bar code object 17

Q

QR Code 28, 29, 30, 52, 54, 57, 58, 93, 96
QR Code special-function parameters 87
quiet zones 10, 18

R

ratio, WE:NE 14, 58
receiver rules 107
reflectivity 14, 15
resolution 14, 20
RM4SCC 28, 29, 30, 44, 54, 56, 57, 58, 93, 95
Royal Mail bar code 44, 95

S

scanner 9, 16
scanning 11
Singapore bar code 44
specification references 109
specification-check exceptions 101
SSCAST flag 65
standard actions, exception conditions 101
start and stop margins 10
start character 10, 18
stop character 10, 18
structured fields, MO:DCA 112
substitution error rate 16
symbol generation, bar code 14
symbol orientation 22
symbol placement 21
symbol size 23
symbologies
BSD modifier field 30
BSD type 28
definition 10
examples 9
generation process 14
symbologies *(continued)*
specification references 109
type 28
syntax diagram iv

T

trademarks 119
trailing blanks adjustment flag 66
transparency 15
two-dimensional matrix symbologies 13
two-dimensional stacked symbologies 13
two-level code 14, 18
types 28

U

UCC/EAN 128 40
unit base 20
units of measure 20
units per unit base 20
UPC 9, 18, 28, 29, 30, 33, 34, 35, 54, 57, 58, 64, 93, 94, 109

W

WE:NE 14, 58
wide-to-narrow ratio 14, 58
Write Bar Code command 114
Write Bar Code Control command 113

X

Xbc, Ybc coordinate system 19

Readers' Comments — We'd Like to Hear from You

Data Stream and Object Architectures
Bar Code Object Content Architecture
Reference

Publication No. S544-3766-05

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

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

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
S544-3766-05

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
IBM Printing Systems Division
Department H7FE, Building 004M
Information Development
P.O. Box 1900
Boulder, CO USA 80301-9817

Fold and Tape | **Please do not staple** | Fold and Tape

S544-3766-05

Cut or Fold
Along Line

Printed in USA

S544-3766-05