Note:
Before using this information, read the information in "Notices" on page 159.

AFPC-0005-09
Tenth Edition (June 2015)

This edition applies to the Bar Code Object Content Architecture™ (BCOCA™). It is the tenth edition of this book and replaces and makes obsolete the previous edition (AFPC-0005-08). This edition remains current until a new edition is published. This publication also applies to any subsequent releases of Advanced Function Presentation™ (AFP™) products that use the BCOCA architecture until otherwise indicated in a new edition.

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 xv.

Internet
Visit our home page: http://www.afpcinc.org
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 Advanced Function Presentation (AFP) environment.

AFP Consortium™ (AFPC™)

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

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

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

The BCOCA Reference was first published by IBM in 1987 as part of the IPDS™ Reference; it was published as an independent architecture document in 1991 and has had several enhancements and updates since that time. The first seven editions were published by IBM Corporation and later editions were published by the AFP Consortium.

First Edition published by IBM Corporation
S544-3766-00 dated August 1991

Second Edition published by IBM Corporation
S544-3766-01 dated July 1993

This edition provides enhanced detail and clarifications:
• Additional information has been provided to aid in the generation of BCOCA objects.
• Chapter 1 has been enhanced to describe how the BCOCA architecture fits into IBM's presentation environments.
• The glossary has been extensively revised.

Third Edition published by IBM Corporation
S544-3766-02 dated December 1997

This edition provides enhanced detail and the following major new functions:
• Additional information to aid in the generation of BCOCA objects
• Check digit details for all symbologies
• Glossary updates
• Many clarifications
• New UPC/EAN supplemental modifiers
• Two new postal bar codes:
  1. Japan Postal Bar Code
  2. Royal Mail Postal Bar Code (RM4SCC)

Fourth Edition published by IBM Corporation
S544-3766-03 dated June 2000

This edition provides enhanced detail and the following major new functions:
• Additional information to aid in the generation of BCOCA objects
• A method of suppressing trailing blanks when bar codes are built from AFP line data
• Editorial improvements for color, module width, bar code descriptions, and the list of symbology specifications
• Information about the Code 39 character set
• Information about UCC/EAN 128
• Two new postal bar codes:
  1. Australia Post Bar Code
  2. Dutch KIX postal bar code (a variation of the RM4SCC code)
Fifth Edition published by IBM Corporation
S544-3766-04 dated May 2001

This edition provides enhanced detail and the following major new functions:
- Additional information to aid in the generation of BCOCA objects
- A method of suppressing a bar code symbol so that just the human-readable interpretation (HRI) is printed
- Three new two-dimensional bar code symbologies:
  1. Data Matrix
  2. MaxiCode
  3. PDF417

Sixth Edition published by IBM Corporation
S544-3766-05 dated November 2003

This edition provides enhanced detail and the following major new functions:
- Additional information, clarifications, and pictures to aid in the generation of BCOCA objects
- Two new bar code types to provide additional symbol variations:
  1. Code 93 1D bar code
  2. QR Code® 2D bar code
- Two new bar code variations:
  1. PLANET, a variation of POSTNET
  2. UCC/EAN 128, a variation of Code 128

Seventh Edition published by IBM Corporation
S544-3766-06 dated July 2006

This edition provides enhanced detail and the following major new functions:
- Additional information, clarifications, and pictures to aid in the generation of BCOCA objects
- A new bar code type:
  - USPS Four-State bar code (also called OneCodeSOLUTION bar code, later renamed to Intelligent Mail® Barcode)
- Enhancements:
  - Additional color spaces (RGB, CMYK, highlight, and CIELAB)
  - Shift-out, shift-in (SOSI) support for QR Code
  - UCC/EAN 128 clarifications and modifier X'04'

Eighth Edition published by the AFP Consortium
AFPC-0005-07 dated January 2011

This edition provides enhanced detail and the following major new functions:
- New bar code types and modifiers:
  - Intelligent Mail Container Barcode
  - Royal Mail RED TAG
- A new BCOCA subset called BCD2
- Enhancements:
  - Clarification for MaxiCode EOT character
  - Control over Data Matrix encodation scheme
  - Correction to Japan Postal check digit algorithm
  - Default parameter value recommendations
  - Desired symbol width parameter
  - GS1 terminology
  - Guidelines for printing HRI
– Retired items identified
– Small fixed-size bar codes
– Small Intelligent Mail Barcodes
– Symbol origin clarification
• Additional information, clarifications, and pictures to aid in the generation of BCOCA objects

Ninth Edition published by the AFP Consortium
AFPC-0005-08 dated May 2012

This edition provides enhanced detail and the following new function:
• A new bar code type and several modifiers for the GS1 DataBar family of bar codes:
  – GS1 DataBar Omnidirectional
  – GS1 DataBar Truncated
  – GS1 DataBar Stacked
  – GS1 DataBar Stacked Omnidirectional
  – GS1 DataBar Limited
  – GS1 DataBar Expanded
  – GS1 DataBar Expanded Stacked
• Bearer Bars for Interleaved 2-of-5 and ITF-14 symbols
• Information about the role of the BCOCA BCD2 subset in MO:DCA™ Interchange Set 3 (IS/3)
• Additional information, clarifications, and pictures to improve readability
How to Use This Book

This book is divided into six chapters and four appendixes:

- **Chapter 1, “A Presentation Architecture Perspective”, on page 1** introduces the AFPC presentation architectures and describes the role of data streams and data objects.
- **Chapter 2, “Introduction to BCOCAs”, on page 7** 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 BCOCAs architecture and its relationship to other presentation architectures.
- **Chapter 4, “BCOCA Data Structures”, on page 29** defines the data structures, fields, and valid data values assigned to and reserved or retired for the BCOCAs architecture.
- **Chapter 5, “Exception Conditions”, on page 131** lists the exceptions to the BCOCAs definitions and what to do when such exceptions occur.
- **Chapter 6, “Compliance”, on page 137** describes requirements for valid generators and receivers of a BCOCAs object.
- **Appendix A, “Bar Code Symbology Specification References”, on page 139** lists the bar code symbology specifications referenced in this document.
- **Appendix B, “MO:DCA Environment”, on page 143** describes how BCOCAs bar code objects are defined and used in the MO:DCA environment.
- **Appendix C, “IPDS Environment”, on page 145** describes how BCOCAs bar code objects are defined and used in the IPDS environment.
- **Appendix D, “Retired Items” on page 151** lists each retired item that is mentioned within the body of this book and also lists those items that have been unretired.

The “Glossary” on page 161 defines terms used within the book.
How to Read the Syntax Diagrams

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

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>The field’s offset, data type, or both</td>
<td>Name of field, if applicable</td>
<td>Range of valid values, if applicable</td>
<td>Meaning or purpose of the data element</td>
<td>Subset of the range of values that must be supported by all BCOCA receivers; refer to Chapter 6, “Compliance”, on page 137 for additional details</td>
</tr>
</tbody>
</table>

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:

Table 2. Bar Code Symbol Data (BSA) Data Structure

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>BITS</td>
<td>Bar code flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 0</td>
<td></td>
<td>HRI</td>
<td>B'0' B'1'</td>
<td>HRI is presented</td>
<td>B'0' B'1'</td>
<td></td>
</tr>
<tr>
<td>bits 1–2</td>
<td></td>
<td>Position</td>
<td>B'00' B'01' B'10'</td>
<td>Default (HRI below)</td>
<td>B'00' B'01' B'10'</td>
<td></td>
</tr>
<tr>
<td>bit 3</td>
<td></td>
<td>SSCAST</td>
<td>B'0' B'1'</td>
<td>Asterisk is not presented</td>
<td>B'0' B'1'</td>
<td></td>
</tr>
<tr>
<td>bit 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>B'0'</td>
<td></td>
</tr>
<tr>
<td>bit 5</td>
<td></td>
<td>Suppress bar code symbol</td>
<td>B'0' B'1'</td>
<td>Bar code suppression: Present symbol Supress symbol</td>
<td>B'0'</td>
<td>B'0' B'1'</td>
</tr>
<tr>
<td>bit 6</td>
<td></td>
<td>Suppress blanks</td>
<td>B'0' B'1'</td>
<td>Desired method of adjusting for trailing blanks: Don't suppress suppressing bar code</td>
<td>B'0'</td>
<td>B'0'</td>
</tr>
<tr>
<td>bit 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1–2</td>
<td>UBIN</td>
<td>X offset</td>
<td>X'0001'–X'7FFF'</td>
<td>X₀X-coordinate of the symbol origin in the bar code presentation space</td>
<td>X'0001'–X'7FFF'</td>
<td>X'0001'–X'7FFF'</td>
</tr>
</tbody>
</table>

Refer to the note following the table.
Table 2  Bar Code Symbol Data (BSA) Data Structure (cont’d.)

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>3–4</td>
<td>UBIN</td>
<td>Y offset</td>
<td>X'0001'–X'7FFF'</td>
<td>Y_{bc}-coordinate of the symbol origin in the bar code presentation space</td>
<td>X'0001'–X'7FFF' Refer to the note following the table.</td>
<td>X'0001'–X'7FFF' Refer to the note following the table.</td>
</tr>
</tbody>
</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 | See field description |

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 | Any value defined for the bar code type selected by the BSD |

**Note:** The BCD1 and BCD2 range for these fields has been specified assuming a unit of measure of 1/1440 of an inch. Many receivers support the BCD1 or BCD2 subset plus additional function. If a receiver supports additional units of measure, the BCOCA architecture requires the receiver to support a range equivalent to the subset 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 21.
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 from left to right beginning with byte 0 with the leftmost byte as most significant; this is called Big Endian. For example, if a structure is three bytes long and has two fields, a two-byte field followed by a one-byte field, the bytes are numbered as follows:
  
  | Bytes 0–1 | Field 1 |
  | Byte 2    | Field 2 |

  Byte 0 is the leftmost, high-order byte for the first field.

- Bit strings are numbered beginning with 0. For example, a one-byte bit string contains bit 0, bit 1, ..., bit 7.
- For numerical binary data, bit 0 is the most significant bit. For example, decimal 13 is equivalent to binary B'00001101'.

- Field values are expressed in hexadecimal or binary notation:
  
  \[ X'7FFF' = +32767 \]
  \[ B'0001' = 1 \]
  \[ B'01111110' = X'7E' = +126 \]

- 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.

- Some fields or values are labeled Retired item n, where n is an identifying number. These fields or values are reserved for a particular purpose and must not be used for any other purpose. Refer to Appendix D. "Retired Items" on page 151 for a description of the individual retired items.

- 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.
### Bar Code Abbreviations

Abbreviations used in this book have the following meanings:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIM USS</td>
<td>Automatic Identification Manufacturers Uniform Symbol Specification</td>
</tr>
<tr>
<td>EAN</td>
<td>European Article Numbering</td>
</tr>
<tr>
<td>GS1</td>
<td>Global Standards 1</td>
</tr>
<tr>
<td>ITF-14</td>
<td>Interleaved 2-of-5 encoding 13 input digits and a check digit</td>
</tr>
<tr>
<td>JAN</td>
<td>Japanese Article Numbering</td>
</tr>
<tr>
<td>MSI</td>
<td>MSI Data Corporation</td>
</tr>
<tr>
<td>PDF417</td>
<td>Portable Data File 417</td>
</tr>
<tr>
<td>PLANET</td>
<td>PostaL Alpha Numeric Encoding Technique (United States Postal Service)</td>
</tr>
<tr>
<td>POSTNET</td>
<td>POSTal Numeric Encoding Technique (United States Postal Service)</td>
</tr>
<tr>
<td>QR Code</td>
<td>Quick Response Code</td>
</tr>
<tr>
<td>RM4SCC</td>
<td>Royal Mail 4 State Customer Code</td>
</tr>
<tr>
<td>UCC</td>
<td>Uniform Code Council</td>
</tr>
<tr>
<td>UPC</td>
<td>Universal Product Code (United States)</td>
</tr>
<tr>
<td>UPC/CGPC</td>
<td>Universal Product Code (United States) and the Canadian Grocery Product Code</td>
</tr>
<tr>
<td>USPS</td>
<td>United States Postal Service</td>
</tr>
<tr>
<td>USS</td>
<td>Uniform Symbol Specification</td>
</tr>
</tbody>
</table>
Related Publications

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

Table 3. AFP Consortium Architecture References

<table>
<thead>
<tr>
<th>AFP Architecture Publication</th>
<th>Book Identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFP Programming Guide and Line Data Reference</td>
<td>S544-3884 (IBM)</td>
</tr>
<tr>
<td>Bar Code Object Content Architecture Reference</td>
<td>AFPC-0005</td>
</tr>
<tr>
<td>Color Management Object Content Architecture Reference</td>
<td>AFPC-0006</td>
</tr>
<tr>
<td>Font Object Content Architecture Reference</td>
<td>S544-3285 (IBM)</td>
</tr>
<tr>
<td>Graphics Object Content Architecture for Advanced Function Presentation Reference</td>
<td>AFPC-0008</td>
</tr>
<tr>
<td>Image Object Content Architecture Reference</td>
<td>AFPC-0003</td>
</tr>
<tr>
<td>Intelligent Printer Data Stream™ Reference</td>
<td>AFPC-0001</td>
</tr>
<tr>
<td>Metadata Object Content Architecture Reference</td>
<td>AFPC-0013</td>
</tr>
<tr>
<td>Mixed Object Document Content Architecture™ (MO:DCA) Reference</td>
<td>AFPC-0004</td>
</tr>
<tr>
<td>Presentation Text Object Content Architecture Reference</td>
<td>SC31-6803 (IBM)</td>
</tr>
</tbody>
</table>

Table 4. Additional AFP Consortium Documentation

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

Table 5. AFP Font-Related Documentation

<table>
<thead>
<tr>
<th>Publication</th>
<th>Book Identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Character Data Representation Architecture Reference and Registry; please refer to the online version for the most current information: <a href="http://www-306.ibm.com/software/globalization/cdra/index.jsp">http://www-306.ibm.com/software/globalization/cdra/index.jsp</a></td>
<td>SC09-2190 (IBM)</td>
</tr>
<tr>
<td>Font Summary for AFP Font Collection</td>
<td>S544-5633 (IBM)</td>
</tr>
<tr>
<td>Technical Reference for Code Pages</td>
<td>S544-3802 (IBM)</td>
</tr>
</tbody>
</table>
Table 6. UP3I™ Architecture Documentation

<table>
<thead>
<tr>
<th>UP3I Publication</th>
<th>Book Identification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Universal Printer Pre- and Post-Processing Interface (UP3I) Specification</td>
<td>Available at <a href="http://www.afpcinc.org">www.afpcinc.org</a></td>
</tr>
</tbody>
</table>
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 enhanced detail to support the BCOCA products that were introduced in the years 2012 through 2015 and to support the work of the AFP Consortium. Specifically, the following new function has been added:

- A new bar code type called Royal Mail Mailmark®
- Two new bar code modifiers for Royal Mail Mailmark: Barcode C (66 bars) and Barcode L (78 bars)
- Royal Mail RED TAG bar code type has been deprecated
- POSTNET and PLANET bar codes have been deprecated
- One new exception ID (EC-1204)
- New appendix describing each numbered retired item and also identifying items that have been unretired
- Metadata Object Content Architecture (MOCA) added; metadata can be carried in MO:DCA print files and documents, but is currently not supported in IPDS data streams
- Extensive glossary additions for color terms and new AFP terms
- Additional information and clarifications to improve readability
# Table of Contents

## Preface
- Who Should Read This Book ................................................................. iii
- AFP Consortium™ (AFPC™) ................................................................. iii
- Publication History ........................................................................ iv
- How to Use This Book .................................................................... vii
- How to Read the Syntax Diagrams ..................................................... viii
- Notation Conventions ...................................................................... viii
- Bar Code Abbreviations ................................................................... xi
- Related Publications ........................................................................ xii

## Changes in This Edition ................................................................. xv

## Figures ............................................................................................. xxiii

## Tables ............................................................................................... xxi

## Chapter 1. A Presentation Architecture Perspective .................... 1
- The Presentation Environment ......................................................... 1
- Architecture Components .............................................................. 2
- Data Streams .................................................................................. 2
- Objects ........................................................................................... 4

## Chapter 2. Introduction to BCCLA .................................................. 7
- What Is a Bar Code? ....................................................................... 7
- How Data Is Presented ................................................................. 7
- How Data Is Retrieved ................................................................. 7
- Elements of a Bar Code System ..................................................... 7
  - Bar Code Symbology .................................................................. 8
  - Linear Symbologies .................................................................... 8
  - Two-Dimensional Matrix Symbologies ....................................... 12
  - Two-Dimensional Stacked Symbologies .................................... 12
  - Bar Code Symbol Generation .................................................. 13
  - Bar Code Encoding Techniques ............................................... 13
  - Information Density .................................................................... 13
  - Physical Media ............................................................................ 14
  - Printers ....................................................................................... 14
  - Scanners .................................................................................... 15
  - Performance Measurement ......................................................... 15

## Chapter 3. BCCLA Overview .......................................................... 17
- General BCCLA Concepts ............................................................ 17
- Bar Code Object Processor ........................................................... 18
- Bar Code Presentation Space .......................................................... 20
- Coordinate System ........................................................................ 20
- Measurements ............................................................................... 20
  - L-unit Range Conversion Algorithm ......................................... 21
  - Symbol Placement ...................................................................... 23
  - Symbol Orientation .................................................................... 24
  - Symbol Size .............................................................................. 25
  - Human-Readable Interpretation (HRI) Guidelines ................. 26

## Chapter 4. BCCLA Data Structures ............................................... 29
- BCD1 Subset .................................................................................. 29
- BCD2 Subset .................................................................................. 29
- Bar Code Symbol Descriptor (BSD) ................................................ 31
- Default Value Recommendations .................................................. 45
- Bar Code Type and Modifier Descriptions .................................. 47
  - Code 39 (3-of-9 Code), AIM USS-39 (modifier values X’01’ and X’02’) ................................................ 47
  - MSI (modified Plessey code, modifier values X’01’ through X’09’) .................................................... 48
  - UPC/CGPC – Version A (modifier value X’00’). .......................... 49

Copyright © AFP Consortium 1991, 2015  xvii
### Appendix D. Retired Items

- BCOCA Exception Conditions and IPDS Exception IDs
- Japan Postal Bar Code (modifier values X'00' and X'01')
- Data Matrix and GS1 DataMatrix (modifier value X'00')
- MaxiCode (modifier value X'00')
- PDF417 (modifier values X'00' and X'01')
- Australia Post Bar Code (modifier values X'01' through X'08')
- QR Code (modifier value X'02')
- Code 93 (modifier value X'00')
- Intelligent Mail Barcode (modifier values X'00' through X'03')
- Royal Mail RED TAG (deprecated), modifier value X'00'
- GS1 DataBar
- Royal Mail Mailmark (modifier values X'00' and X'01')
- Check Digit Calculation Methods
- Bar Code Symbol Data (BSA)
- Data Matrix Special-Function Parameters
- MaxiCode Special-Function Parameters
- PDF417 Special-Function Parameters
- QR Code Special-Function Parameters
- Valid Code Pages and Type Styles
- Valid Characters and Data Lengths
- Characters and Code Points
- Code 128 Code Page

### Chapter 6. Compliance

- Generator Rules
- Receiver Rules

### Appendix A. Bar Code Symbology Specification References

- UPC/CGPC – Version E (modifier value X'00')
- UPC – Two-Digit Supplemental (modifier values X'00' through X'02')
- UPC – Five-Digit Supplemental (modifier values X'00' through X'02')
- EAN-8 (includes JAN-short, modifier value X'00')
- EAN-13 (includes JAN-standard, modifier value X'00')
- Industrial 2-of-5 (modifier values X'01' and X'02')
- Matrix 2-of-5 (modifier values X'01' and X'02')
- Interleaved 2-of-5, ITF-14, AIM USS-1 2/5 (modifier values X'01' through X'04')
- Codabar, 2-of-7, AIM USS-Codabar (modifier values X'01' and X'02')
- Code 128 (modifier values X'02' through X'05')
  - Code 128 modifier X'02' – Code 128 symbol, using original (1986) start-character algorithm
  - Code 128 modifier X'03' – UCC/EAN 128 symbol, without parentheses in the HRI
  - Code 128 modifier X'04' – UCC/EAN 128 and GS1-128 symbols, with parentheses in the HRI
- Code 128 modifier X'05’ – Intelligent Mail Container Barcode
- EAN Two-Digit Supplemental (modifier values X'00' and X'01')
- EAN Five-Digit Supplemental (modifier values X'00' and X'01')
- POSTNET and PLANET (both deprecated, modifier values X'00' through X'04')
- Royal Mail RM4SCC and Dutch KIX (modifier values X'00' and X'01')
- Data Matrix and GS1 DataMatrix (modifier value X'00')
- MaxiCode (modifier value X'00')
- PDF417 (modifier values X'00' and X'01')
- Australia Post Bar Code (modifier values X'01' through X'08')
- QR Code (modifier value X'02')
- Code 93 (modifier value X'00')
- Intelligent Mail Barcode (modifier values X'00' through X'03')
- Royal Mail RED TAG (deprecated), modifier value X'00'
- GS1 DataBar
- Royal Mail Mailmark (modifier values X'00' and X'01')

### Chapter 5. Exception Conditions

- Specification-Check Exceptions
- Data-Check Exceptions

### Appendix B. MO:DCA Environment

- Bar Codes in MO:DCA Documents
- Bar Code Data Object Structured Fields
- Bar Code Data Descriptor (BDD)
- Bar Code Data (BDA)

### Appendix C. IPDS Environment

- IPDS Bar Code Command Set
- Write Bar Code Control Command
- Write Bar Code Command
- Additional Related Commands
- BCOCA Exception Conditions and IPDS Exception IDs

### Appendix D. Retired Items

- Retired Items

---

**BCOCA Reference**
<table>
<thead>
<tr>
<th>Notices</th>
<th>159</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trademarks</td>
<td>160</td>
</tr>
<tr>
<td>Glossary</td>
<td>161</td>
</tr>
<tr>
<td>Index</td>
<td>197</td>
</tr>
</tbody>
</table>
Figures

1. Presentation Environment ................................................................................................................1
2. Presentation Model ..........................................................................................................................3
3. Presentation Page ...........................................................................................................................5
4. Bar Code Symbol Structure ...........................................................................................................8
5. Examples of Linear Bar Code Symbols (spans three pages) ..........................................................9
6. Examples of 2D Matrix Bar Code Symbols ....................................................................................12
7. Examples of 2D Stacked Bar Code Symbols ................................................................................12
8. Bar Code Presentation Space .........................................................................................................20
10. BCOD Function and Subsetting ....................................................................................................30
11. Example of a MaxiCode Bar Code Symbol with Zipper and Contrast Block .........................104
12. Subset of EBCDIC Code Page 500 That Can Be Translated To GLI 0 ........................................106
13. Subset of EBCDIC Code Page 500 That Can Be Translated To ECI 000020 .............................114
## Tables

1. Data Structure Syntax .......................................................................................................................... viii
2. Bar Code Symbol Data (BSA) Data Structure ...................................................................................... viii
3. AFP Consortium Architecture References ......................................................................................... xii
4. Additional AFP Consortium Documentation ....................................................................................... xii
5. AFP Font-Related Documentation ..................................................................................................... xii
6. UP3™ Architecture Documentation ................................................................................................... xiii
7. Field Ranges for Commonly-Supported Measurement Bases ............................................................ 22
8. Human-Readable Interpretation Type Style Recommendations ....................................................... 26
10. Bar Code Types ............................................................................................................................... 34
11. Modifier Values by Bar Code Type .................................................................................................... 36
12. Standard OCA Color-Value Table ...................................................................................................... 39
13. Sizing Targets for Fixed-Size Bar Code Types .................................................................................. 41
14. Recommended Default Values for Module Width, Element Height, and Wide-to-Narrow Ratio ...... 45
15. Intelligent Mail Container Barcode Data Field Ranges ................................................................... 60
16. Valid Code Points for Direct Input to a Japan Postal Bar Code ........................................................... 67
17. Australia Post Modifier Values ......................................................................................................... 72
18. Royal Mail RED TAG (deprecated) Data Field Ranges ................................................................. 76
19. Modifier Values for a GS1 DataBar Expanded Stacked Bar Code .................................................... 81
20. Check Digit Calculation Methods .................................................................................................... 84
22. Data Matrix Special-Function Parameters ......................................................................................... 94
23. Supported Sizes for a Data Matrix Symbol ....................................................................................... 96
24. MaxiCode Special-Function Parameters ......................................................................................... 100
25. PDF417 Special-Function Parameters ............................................................................................... 105
26. QR Code Special-Function Parameters ......................................................................................... 111
27. Supported Versions for a QR Code Symbol .................................................................................... 115
28. Valid Code Pages and Type Styles .................................................................................................. 118
29. Valid Characters and Data Lengths .................................................................................................. 120
30. Characters and Code Points Commonly used in the BCOCA Symbologies (Not a Complete Listing) 126
31. MO:DCABar Code Data Descriptor (BDD). ...................................................................................... 144
32. MO:DCABar Code Data (BDA) ........................................................................................................ 144
33. IPDS Bar Code Data Descriptor (BCDD) ......................................................................................... 146
34. BCOCA Exception Conditions and IPDS Exception IDs ................................................................. 148
35. Valid Code Pages and Type Styles .................................................................................................. 156
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. AFP 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. AFP presentation architectures provide structures that support object-oriented models and client/server environments.

AFP 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, AFP presentation architectures use industry and international standards, such as the ITU-TSS (formerly known as CCITT) facsimile standards for compressed image data.
Architecture Components

AFP 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.

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.
Figure 2 shows a system model relating MO:DCA and IPDS data streams to the presentation environment previously described. Also shown in the model are the object content architectures that 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, that can be individually positioned and manipulated to meet the needs of the presentation application.

The AFPC-defined object content architectures are:

- **Presentation Text Object Content Architecture (PTOCA)**: A data architecture for describing text objects that have been formatted for all-points-addressable presentations. Specifications of fonts, text color, and other visual attributes are included in the architecture definition.

- **Image Object Content Architecture (IOCA)**: A data architecture for describing resolution-independent image objects captured from a number of different sources. Specifications of recording formats, data compression, color, and grayscale encoding are included in the architecture definition.

- **Graphics Object Content Architecture for Advanced Function Presentation (AFP GOCA)**: A version of GOCA that is used in Advanced Function Presentation (AFP) environments. GOCA is 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.

- **Bar Code Object Content Architecture (BCOCA)**: A data architecture for describing bar code objects, using a number of different symbologies. Specification of the data to be encoded and the symbology attributes to be used are included in the architecture definition.

- **Font Object Content Architecture (FOCA)**: A resource architecture for describing the structure and content of fonts referenced by presentation data objects in the document.

- **Color Management Object Content Architecture (CMOCA)**: A resource architecture used to carry the color management information required to render presentation data.

- **Metadata Object Content Architecture (MOCA)**: A resource architecture used to carry metadata in an AFP environment.

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

In addition to 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.
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.
Elements of a Bar Code System

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 that might 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 139 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 can 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.

Figure 5. Examples of Linear Bar Code Symbols (spans three pages)

(Part 1 of figure)
Elements of a Bar Code System

(Part 2 of figure)
Elements of a Bar Code System

(Part 3 of figure)

GS1 DataBar Omnidirectional
(encoding 20012345678909)

GS1 DataBar Truncated
(encoding 20012345678909)

GS1 DataBar Stacked
(encoding 20012345678909)

GS1 DataBar Stacked Omnidirectional
(encoding 20012345678909)

GS1 DataBar Limited
(encoding 15012345678907)

GS1 DataBar Expanded
(encoding 0198898765432106320201234515991231)

GS1 DataBar Expanded Stacked
(encoding 0198898765432106320201234515991231)
Elements of a Bar Code System

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 most 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.

Figure 6. Examples of 2D Matrix Bar Code Symbols

![Data Matrix 2D Symbol](encoding A1H2C3D4E5F6G7H8I9J0K1L2) ![MaxiCode 2D Symbol] ![QR Code 2D Symbol]

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. Examples of 2D Stacked Bar Code Symbols

![PDF417 Truncated PDF417](Start Pattern, 3 Data Symbol Characters per Row, 13 Rows)
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 code 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 can 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 can 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 can 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 can 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, that 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:

- An 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
- An environment that controls how line data is presented; for example, the AFP Line Data 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 143 and Appendix C, “IPDS Environment”, on page 145. When multiple bar code symbols of the same type are to be printed on a page, the symbols can be combined into a single object, which avoids having to repeat the same descriptor in multiple objects.
Bar Code Object Processor

A BCODCA 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 20. 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 143 and Appendix C, “IPDS Environment”, on page 145.

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 (or desired symbol width)
• 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 (that 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 and HRI information is totally contained within the bar code presentation space, and that there is sufficient empty space for the quiet zones.

Notes:

1. The BCOCA object generator is responsible for insuring that there is sufficient empty space for 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 7 on page 22 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” that in hexadecimal is X'0001' to X'1555'. Table 7 shows the BCOCA-required ranges for several commonly supported measurement bases.

Table 7. Field Ranges for Commonly-Supported Measurement Bases

<table>
<thead>
<tr>
<th>14400 units per 10 inches</th>
<th>5670 units per 10 centimeters</th>
<th>2400 units per 10 inches</th>
<th>945 units per 10 centimeters</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0001'–X'7FFF'</td>
<td>X'0001'–X'7FFF'</td>
<td>X'0001'–X'1555'</td>
<td>X'0001'–X'1555'</td>
</tr>
</tbody>
</table>
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 corner of an imaginary rectangle of minimum size that bounds the bar-space patterns (or two-dimensional module patterns) 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.

**Note:** In most cases, the symbol origin is the top-left corner of the leftmost bar; however, this is not an appropriate origin for some bar code types, such as Dutch KIX, Intelligent Mail Barcode, MaxiCode, and Royal Mail Mailmark. The original BCOCA symbol origin definition was the “top-left corner of the leftmost bar”; therefore, some implementations might still use the original definition (this is not considered to be a deviation from the architecture for these older implementations). For GS1 DataBar symbols, the origin of the bar code symbol is the top-left corner of the leftmost space (since GS1 DataBar symbols begin with a space).

The BCOCA object generator is responsible for insuring that there is sufficient empty space for 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, the Bearer Bars (Interleaved 2-of-5), the Identification Bars and USPS Banner (Intelligent Mail Container Barcode), the RED TAG indicator (Royal Mail RED TAG (deprecated)), the zipper pattern and contrast block (MaxiCode), and the HRI, extends outside of the bar code presentation space.
Symbol Orientation

Bar code users typically think of a bar code symbol in one of two orientations (picket fence or ladder) although linear symbols are usually defined in the picket fence orientation. Orientation of a bar code symbol into either the picket fence or ladder orientation is accomplished by rotating the bar code object area 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.

All BCODA implementations allow the object area to be rotated to one or more of the following orientations: 0°, 90°, 180°, 270°. Most of the implementations support all four orientations, thus allowing a bar code symbol to be presented in either a picket fence or ladder orientation or in one of the other two (upside-down) orientations. In addition, some BCODA implementations allow the object area to be rotated to any angle.

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 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 BCODA parameters. The width of the symbol is usually dependent on the amount of data to be encoded and by choices made in various BCODA parameters. Default values exist for most of the BCODA 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 BCODA printers.

Some BCODA implementations support the desired symbol width parameter. This parameter provides a target width for the symbol and allows the BCODA receiver to calculate an optimal module width value based on the target width. Implementations that don’t support the desired symbol width parameter require the BCODA generator to provide an appropriate module width value.

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 42 for a description of the height for specific linear symbols. Some bar code symbologies explicitly specify the bar code symbol height; in this case, the element-height and height-multiplier parameters are ignored. The symbologies that explicitly specify the symbol height are as follows: Australia Post Bar Code, Intelligent Mail Barcode, Japan Postal Bar Code, POSTNET (deprecated), RM4SCC, Royal Mail RED TAG (deprecated), and Royal Mail Mailmark.

Two-Dimensional Matrix Symbologies

The MaxiCode symbology specifies a fixed physical size; the element-height and height-multiplier parameters are ignored for MaxiCode symbols. Some BCODA receivers provide "small-symbol support" that allows the symbol to be produced at either an optimal or a small size; the module-width parameter is used to select the small or optimal size.

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.
Bar code symbols are meant to be read by machines and are usually difficult for a human to interpret; therefore some bar code symbols allow a human-readable interpretation (HRI) to be printed near the symbol. HRI is 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. The BCoca architecture allows the bar code object to specify whether or not HRI is printed and whether the HRI is above or below the symbol. Table 8 shows which bar code types allow HRI and recommends a font type style for each.

The first place a BCoca implementor should look for HRI guidelines is the bar code symbology specification; if the symbology specification does not provide enough details on HRI, the implementor should then use the BCoca guidelines described in this section.

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>HRI Supported?</th>
<th>Recommended Font Type Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Code 39 (3-of-9 Code), AIM USS-39</td>
<td>Yes; above or below</td>
<td>OCR A</td>
</tr>
<tr>
<td>X'02'</td>
<td>MSI (modified Plessey code)</td>
<td>Yes; above or below</td>
<td>OCR A</td>
</tr>
<tr>
<td>X'03'</td>
<td>UPC/CGPC – Version A</td>
<td>Yes; below only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'05'</td>
<td>UPC/CGPC – Version E</td>
<td>Yes; below only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'06'</td>
<td>UPC – Two-Digit Supplemental (Periodicals)</td>
<td>Yes; above only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'07'</td>
<td>UPC – Five-Digit Supplemental (Paperbacks)</td>
<td>Yes; above only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'08'</td>
<td>EAN-8 (includes JAN-short)</td>
<td>Yes; below only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'09'</td>
<td>EAN-13 (includes JAN-standard)</td>
<td>Yes; below only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'0A'</td>
<td>Industrial 2-of-5</td>
<td>Yes; above or below</td>
<td>OCR A</td>
</tr>
<tr>
<td>X'0B'</td>
<td>Matrix 2-of-5</td>
<td>Yes; above or below</td>
<td>OCR A</td>
</tr>
<tr>
<td>X'0C'</td>
<td>Interleaved 2-of-5, ITF-14, AIM USS-I 2/5</td>
<td>Yes; above or below</td>
<td>OCR A</td>
</tr>
<tr>
<td>X'0D'</td>
<td>Codabar, 2-of-7, AIM USS-Codabar</td>
<td>Yes; above or below</td>
<td>OCR A</td>
</tr>
<tr>
<td>X'11'</td>
<td>Code 128, AIM USS-128 Code 128 modifier X'02'</td>
<td>Yes; above or below</td>
<td>OCR B</td>
</tr>
<tr>
<td></td>
<td>UCC/EAN 128 Code 128 modifier X'03'</td>
<td>Yes; above or below</td>
<td>OCR B</td>
</tr>
<tr>
<td></td>
<td>UCC/EAN 128 and GS1-128 Code 128 modifier X'04'</td>
<td>Yes; above or below</td>
<td>OCR B</td>
</tr>
<tr>
<td></td>
<td>Intelligent Mail Container Barcode Code 128 modifier X'05'</td>
<td>Yes; below only</td>
<td>a bold, sans-serif font</td>
</tr>
<tr>
<td>X'16'</td>
<td>EAN Two-Digit Supplemental</td>
<td>Yes; above only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'17'</td>
<td>EAN Five-Digit Supplemental</td>
<td>Yes; above only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'18'</td>
<td>POSTNET (deprecated) and PLANET (deprecated)</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>X'1A'</td>
<td>RM4SCC and Dutch KIX</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>X'1B'</td>
<td>Japan Postal Bar Code</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>X'1C'</td>
<td>Data Matrix, GS1 DataMatrix (2D bar code)</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>X'1D'</td>
<td>MaxiCode (2D bar code)</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>Type</td>
<td>Bar Code Symbology</td>
<td>HRI Supported?</td>
<td>Recommended Font Type Style</td>
</tr>
<tr>
<td>------</td>
<td>------------------------------------</td>
<td>----------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>X'1E'</td>
<td>PDF417 (2D bar code)</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>X'1F'</td>
<td>Australia Post Bar Code</td>
<td>Yes; above only</td>
<td>OCR A</td>
</tr>
<tr>
<td>X'20'</td>
<td>QR Code (2D bar code)</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>X'21'</td>
<td>Code 93</td>
<td>Yes; above or below</td>
<td>OCR B plus the □ and ■ characters</td>
</tr>
<tr>
<td>X'22'</td>
<td>Intelligent Mail Barcode</td>
<td>Yes; above or below</td>
<td>OCR A</td>
</tr>
<tr>
<td>X'23'</td>
<td>Royal Mail RED TAG (deprecated)</td>
<td>No</td>
<td>None</td>
</tr>
<tr>
<td>X'24'</td>
<td>GS1 DataBar</td>
<td>Yes; below only</td>
<td>OCR B</td>
</tr>
<tr>
<td>X'25'</td>
<td>Royal Mail Mailmark</td>
<td>No</td>
<td>None</td>
</tr>
</tbody>
</table>

The Bar Code Symbol Data (BSA) structure contains flags (in byte 0) that control whether or not HRI is printed (bit 0), for some symbols whether the HRI is positioned above or below the symbol (bits 1–2), and for Code 39 symbols whether or not a asterisk is presented for the start and stop characters (bit 3). These flags are ignored for symbologies that do not allow HRI or that explicitly specify the presence and position of the HRI. 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'.

The Bar Code Symbol Descriptor (BSD) structure contains 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 symbologies, such as UPC, EAN, and Intelligent Mail Barcode specify the size and position of the HRI characters. Other symbologies do not provide guidance; for these it is recommended that the font size be selected based on the width of the bar code symbol and that the HRI string be centered on the width of the bar code symbol. It is also recommended that the distance between the characters and the bars be one module width.

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/CGPC Version E, the check digit is always presented in the bar code bar and space patterns, but is not always presented in the HRI. Refer to "Check Digit Calculation Methods" on page 84 for a description of check digit calculation methods and the presence or absence of the check digit in the HRI.

Code 128 modifier X'04' causes left and right parentheses to be shown within the HRI string to distinguish each application identifier within a GS1-128 symbol. Application identifiers are also surrounded by parentheses in the HRI for GS1 DataBar symbols.
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 that is shown in the Range column of the syntax table; the \textit{BCD1 Range} column specifies the values that every receiver supports. Most receivers support more than the minimum ranges.

BCD2 Subset

BCD2 is a superset of BCD1 that provides additional function and bar code types that are required by the MO:DCA IS/3 interchange set. In particular, BCD2 adds the following functions:

- Additional bar code types:
  - Australia Post Bar Code
  - Codabar
  - Code 93
  - Code 128, modifiers X'02' and X'03'
  - Data Matrix (2D bar code)
  - Intelligent Mail Barcode
  - Japan Postal Bar Code
  - MaxiCode (2D bar code)
  - PDF417 (2D bar code)
  - QR Code (2D bar code)
  - RM4SCC (Royal Mail and Dutch KIX)

- Bar code symbol suppression
- Color specification triplet in the MO:DCA and IPDS Bar Code Data Descriptor
- Full range for font local IDs
- Full range for units per unit base

The AFP Consortium recommends that BCOCA implementations support at least the function defined for BCD2.
Figure 10. BCOCA Function and Subsetting

**BCOCA**

**Bar Code Types and Modifiers:**
- Bearer Bars - Interleaved 2-of-5, ITF-14, AIM USS-I 2/5 modifiers X'03' and X'04'
- Code 128 - GS1-128, UCC/EAN 128 modifier X'04'
- Code 128 - Intelligent Mail Container Barcode modifier X'05'
- EAN Five-digit Supplemental - modifier X'01'
- EAN Two-digit Supplemental - modifier X'01'
- GS1 Data Bar
- Industrial 2-of-5
- Matrix 2-of-5
- POSTNET (deprecated) - modifiers X'00' through X'03'
- POSTNET (deprecated) - PLANET (deprecated) modifier X'04'
- Royal Mail Mailmark
- Royal Mail RED TAG (deprecated)
- UPC Two-digit Supplemental - modifiers X'01' and X'02'
- UPC Five-digit Supplemental - modifiers X'01' and X'02'

**Additional Function:**
- 0°, 90°, 180°, and 270° object-area orientation support
- All values of degrees and minutes for object-area orientation
- Bar code objects may be sent in any order
- Desired method of adjusting for trailing blanks
- Desired symbol width
- Small-symbol support
- Standard OCA color support
- User control of Data Matrix encodation scheme

**BCD2**

**Bar Code Types and Modifiers:**
- Australia Post Bar Code
- Codabar, 2-of-7, AIM USS-Codabar
- Code 93
- Code 128 - AIM USS-128 modifier X'02'
- Code 128 - UCC/EAN 128 modifier X'03'
- Data Matrix, GS1 DataMatrix (2D bar code)
- Intelligent Mail Barcode
- Japan Postal Bar Code
- MaxiCode (2D bar code)
- PDF417 (2D bar code)
- QR Code (2D bar code)
- RM45CC - modifier X'00'
- RM45CC - Dutch KIX modifier X'01'

**Additional Function:**
- Extended bar code color support
- Full range for font local IDs
- Full range for units per unit base
- Symbol suppression

**BCD1**

**Bar Code Types and Modifiers:**
- EAN 8 (includes JAN-short)
- EAN 13 (includes JAN-standard)
- EAN Five-digit Supplemental - modifier X'00'
- EAN Two-digit Supplemental - modifier X'00'
- Interleaved 2-of-5, ITF-14, AIM USS-I 2/5 - modifiers X'01' and X'02'
- MSI (modified Plessey code)
- UPC/CGPC Version A
- UPC/CGPC Version E
- UPC Five-digit Supplemental - modifier X'00'
- UPC Two-digit Supplemental - modifier X'00'

**Additional Function:**
- Zero-degree object-area orientation support

---

**Key:**
- BCOCA
- BCD2
- BCD1
### 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.

**Table 9. Bar Code Symbol Descriptor (BSD) Data Structure**

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>CODE</td>
<td>Unit base</td>
<td>X’00’-X’01’</td>
<td>Ten inches Ten centimeters</td>
<td>X’00’</td>
<td>X’00’</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>X’00’</td>
<td>Reserved</td>
<td>X’00’</td>
<td>X’00’</td>
</tr>
<tr>
<td>2–3</td>
<td>UBIN</td>
<td>XUPUB</td>
<td>X’0001’-X’7FFF’</td>
<td>Units per unit base in the (X_{bc}) direction</td>
<td>X’3840’</td>
<td>X’0001’-X’7FFF’</td>
</tr>
<tr>
<td>4–5</td>
<td>UBIN</td>
<td>YUPUB</td>
<td>X’0001’-X’7FFF’</td>
<td>Units per unit base in the (Y_{bc}) direction; must be the same as XUPUB</td>
<td>X’3840’</td>
<td>X’0001’-X’7FFF’</td>
</tr>
<tr>
<td>6–7</td>
<td>UBIN</td>
<td>X extent</td>
<td>X’0001’-X’7FFF’</td>
<td>Width of bar code presentation space in L-units</td>
<td>X’0001’-X’7FFF’</td>
<td>X’0001’-X’7FFF’</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X’FFFF’</td>
<td>Default</td>
<td>X’FFFF’</td>
<td></td>
</tr>
<tr>
<td>8–9</td>
<td>UBIN</td>
<td>Y extent</td>
<td>X’0001’-X’7FFF’</td>
<td>Length of bar code presentation space in L-units</td>
<td>X’0001’-X’7FFF’</td>
<td>X’0001’-X’7FFF’</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X’FFFF’</td>
<td>Default</td>
<td>X’FFFF’</td>
<td></td>
</tr>
<tr>
<td>10–11</td>
<td>UBIN</td>
<td>Symbol width</td>
<td>X’0000’</td>
<td>Desired symbol width: Not specified (use module width)</td>
<td>X’0000’</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X’0001’-X’7FFF’</td>
<td>Desired width of symbol in L-units</td>
<td>X’0000’</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X’FFFF’</td>
<td>Not supported by all BCOCA receivers</td>
<td>X’FFFF’</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>CODE</td>
<td>Type</td>
<td>X’01’-X’03’, X’05’-X’0D’, X’11’, X’16’-X’18’, X’1A’-X’25’</td>
<td>Bar code type</td>
<td>Specified in Table 10 on page 34</td>
<td>Specified in Table 10 on page 34</td>
</tr>
<tr>
<td>13</td>
<td>CODE</td>
<td>Modifier</td>
<td>See field description</td>
<td>Bar code modifier</td>
<td>Specified in Table 11 on page 36</td>
<td>Specified in Table 11 on page 36</td>
</tr>
<tr>
<td>14</td>
<td>CODE</td>
<td>Local ID</td>
<td>X’00’-X’FE’</td>
<td>Font Local ID for HRI Default</td>
<td>X’01’-X’7F’, X’FF’</td>
<td>X’00’-X’FE’, X’FF’</td>
</tr>
<tr>
<td>15–16</td>
<td>CODE</td>
<td>Color</td>
<td>X’0000’-X’0010’, X’FF00’-X’FF08’, X’FFFF’</td>
<td>Color</td>
<td>X’FF07’</td>
<td>X’FF07’</td>
</tr>
<tr>
<td>17</td>
<td>UBIN</td>
<td>Module width</td>
<td>X’01’-X’FE’, X’FF’</td>
<td>Module width in mils Default</td>
<td>Device specific X’FF’</td>
<td>Device specific X’FF’</td>
</tr>
</tbody>
</table>
Bar Code Symbol Descriptor (BSD)

Table 9  Bar Code Symbol Descriptor (BSD) Data Structure (cont’d.)

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>18–19</td>
<td>UBIN</td>
<td>Element height</td>
<td>X'0001'–X'FFFF'</td>
<td>Element height in L-units</td>
<td>X'0001'–X'7FFF'</td>
<td>X'0001'–X'7FFF'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'FFFF'</td>
<td>Default</td>
<td>X'FFFF'</td>
<td>X'FFFF'</td>
</tr>
<tr>
<td>20</td>
<td>UBIN</td>
<td>Multiplier</td>
<td>X'01'–X'FF'</td>
<td>Height multiplier</td>
<td>X'01'–X'FF'</td>
<td>X'01'–X'FF'</td>
</tr>
<tr>
<td>21–22</td>
<td>UBIN</td>
<td>WE:NE</td>
<td>X'0000'</td>
<td>Bar code (see byte 12) does not use ratio</td>
<td>X'0000'</td>
<td>X'0000'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'0001'–X'FFFF'</td>
<td>Wide-to-narrow ratio</td>
<td>At least one value</td>
<td>At least one value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'FFFF'</td>
<td>Default</td>
<td>X'FFFF'</td>
<td>X'FFFF'</td>
</tr>
</tbody>
</table>

Note: The BCD1 and BCD2 range for these fields has been specified assuming a unit of measure of 1/1440 of an inch. Many receivers support the BCD1 or BCD2 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 subset 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 21.

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**

Unit base

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.

The value X'02' is retired as Retired item 1.

**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**

X extent

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  

Y extent  

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  

Desired symbol width (not supported by all BCOCA receivers)  

**Note:** This is an optional parameter that is not supported by all BCOCA receivers; this parameter is ignored by products that do not support this function. IPDS printers report support for this function with property pair X'1302'.

Specifies a desired width for the entire bar code symbol in L-units. The measurement base is specified in bytes 0–5. A value of X'0000' indicates that the width of the symbol is determined by other BSD parameters (module width, WE:NE, and amount of data). For BCOCA receivers that support the desired symbol width parameter, exception condition EC-0610 exists if the specified value is invalid.

The quiet zone is not included in the symbol width for most bar code types. However, when Bearer Bars are used with an Interleaved 2-of-5 symbol, the symbol width includes the quiet zone on both ends of the symbol and also the width of the vertical Bearer Bars (if present).

The BCOCA receiver will use the desired symbol width value to attempt to create the widest bar code symbol that fits within the desired symbol width. The BCOCA receiver does this by:

1. Ignoring the specified module width value (byte 17)
2. Calculating an optimal module width value that will produce the widest symbol that fits into the desired width. The following algorithm is used for all symbologies except for fixed-size symbols:
   a. First the BCOCA receiver calculates how many $X$ values there will be in the symbol and divides this total into the desired symbol width producing a target $X$ value. $X$ is the term used to describe the intended width of a bar code's narrowest element (a bar or a 2D module; spaces are also measured in $X$ values). Wide elements are multiples of the narrow element. For symbologies that use a wide-to-narrow ratio (WE:NE), the multiple is not necessarily an integer value.
   b. Then the target value is converted into printer pel units and adjusted by rounding down to the nearest pel. If the result is larger than the maximum supported module width, the maximum supported module width is used.
      Exception EC-0611 exists if the result is smaller than the minimum supported module width. The standard action for this exception condition is to produce a bar code symbol using the module width value (byte 17); this symbol will be larger than the desired symbol width.
   c. The resulting value replaces the module width value within the BSD and the symbol is generated using that value and all of the other user-specified BSD values to produce the requested symbol. The resulting symbol might be smaller than the desired symbol width.
3. For fixed-size symbols, the optimal-symbol-size value is used unless the BCOCA receiver provides small-symbol support (in which case the value used can be either the optimal or the small value, whichever is best for producing a symbol close to the desired symbol width). Exception condition EC-0611 exists if the resulting fixed-size symbol is wider than the desired symbol width.

The fixed-size bar code types are: Australia Post Bar Code, Dutch KIX, Intelligent Mail Barcode, MaxiCode, PLANET (deprecated), POSTNET (deprecated), RM4SCC, Royal Mail RED TAG (deprecated), and Royal Mail Mailmark.
For UPC or EAN symbols that include a supplemental (bar code types X'06', X'07', X'16', X'17' with modifier X'01' or X'02'), the desired symbol width includes both the base symbol and the supplemental.

**Note:** When a non-zero value is specified in the desired-symbol-width field, an appropriate module-width value should also be specified in byte 17 (a good choice is X'FF' to select the default module width). The module-width value is used in the following cases:

- When the standard action for exception EC-0611 is taken because the printer cannot generate a symbol that fits within the desired width.
- When the bar code object is sent to a BCoca receiver that does not support the desired-symbol-width parameter.
- When X'0000' is specified in the desired-symbol-width field.

### 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 bar code presentation space, 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:

**Table 10. Bar Code Types**

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>In BCD1 Subset?</th>
<th>In BCD2 Subset?</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Code 39 (3-of-9 Code), AIM USS-39</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'02'</td>
<td>MSI (modified Plessey code)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'03'</td>
<td>UPC/CGPC—Version A</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'05'</td>
<td>UPC/CGPC—Version E</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'06'</td>
<td>UPC—Two-Digit Supplemental (Periodicals)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'07'</td>
<td>UPC—Five-Digit Supplemental (Paperbacks)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'08'</td>
<td>EAN-8 (includes JAN-short)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'09'</td>
<td>EAN-13 (includes JAN-standard)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'0A'</td>
<td>Industrial 2-of-5</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'0B'</td>
<td>Matrix 2-of-5</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'0C'</td>
<td>Interleaved 2-of-5, ITF-14, AIM USS-I 2/5</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'0D'</td>
<td>Codabar, 2-of-7, AIM USS-Codabar</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'11'</td>
<td>Code 128, GS1-128, UCC/EAN 128, AIM USS-128, Intelligent Mail Container Barcode</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'16'</td>
<td>EAN Two-Digit Supplemental</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'17'</td>
<td>EAN Five-Digit Supplemental</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'18'</td>
<td>POSTNET (deprecated) and PLANET (deprecated)</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'1A'</td>
<td>RM4SCC and Dutch KIX</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'1B'</td>
<td>Japan Postal Bar Code</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Type</td>
<td>Bar Code Symbology</td>
<td>In BCD1 Subset?</td>
<td>In BCD2 Subset?</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------------------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>X'1C'</td>
<td>Data Matrix, GS1 DataMatrix (2D bar code)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'1D'</td>
<td>MaxiCode (2D bar code)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'1E'</td>
<td>PDF417 (2D bar code)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'1F'</td>
<td>Australia Post Bar Code</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'20'</td>
<td>QR Code (2D bar code)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'21'</td>
<td>Code 93</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'22'</td>
<td>Intelligent Mail Barcode</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'23'</td>
<td>Royal Mail RED TAG (deprecated)</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'24'</td>
<td>GS1 DataBar</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'25'</td>
<td>Royal Mail Mailmark</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Retired Bar Code Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Retired Item</th>
<th>In BCD1 Subset?</th>
<th>In BCD2 Subset?</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'04'</td>
<td>Retired item 7</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'0E'</td>
<td>Retired item 10</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'0F'</td>
<td>Retired item 11</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'10'</td>
<td>Retired item 12</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'12'</td>
<td>Retired item 13</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'13'</td>
<td>Retired item 14</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'14'</td>
<td>Retired item 15</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'15'</td>
<td>Retired item 16</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'19'</td>
<td>Retired item 19</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'EC'</td>
<td>Retired item 22</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'ED'</td>
<td>Retired item 23</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'EE'</td>
<td>Retired item 24</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'EF'</td>
<td>Retired item 25</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Bar Code Symbol Descriptor (BSD)

**Modifier**

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 "Check Digit Calculation Methods" on page 84. Exception condition EC-0B00 exists if the bar code modifier is invalid or unsupported for the bar code type specified.

**Table 11** defines the BCD1 and BCD2 bar code modifier codes that must be supported for each bar code type specified.

**Table 11. Modifier Values by Bar Code Type**

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Modifier Value (byte 13)</th>
<th>In BCD1 Subset?</th>
<th>In BCD2 Subset?</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Code 39 (3-of-9 Code), AIM USS-39</td>
<td>X'01' and X'02'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'02'</td>
<td>MSI (modified Plessey code)</td>
<td>X'01' through X'09'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'03'</td>
<td>UPC/CGPC Version A</td>
<td>X'00'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'05'</td>
<td>UPC/CGPC Version E</td>
<td>X'00'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'06'</td>
<td>UPC - Two-Digit Supplemental</td>
<td>X'00'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'01' and X'02'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'07'</td>
<td>UPC - Five-Digit Supplemental</td>
<td>X'00'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'01' and X'02'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'08'</td>
<td>EAN 8 (includes JAN-short)</td>
<td>X'00'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'09'</td>
<td>EAN 13 (includes JAN-standard)</td>
<td>X'00'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>X'0A'</td>
<td>Industrial 2-of-5</td>
<td>X'01' and X'02'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'0B'</td>
<td>Matrix 2-of-5</td>
<td>X'01' and X'02'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'0C'</td>
<td>Interleaved 2-of-5, ITF-14, AIM USS-I 2/5</td>
<td>X'01' through X'02'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'03' through X'04'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'0D'</td>
<td>Codabar, 2-of-7, AIM USS-Codabar</td>
<td>X'01' and X'02'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'11'</td>
<td>Code 128, UCC/EAN 128, AIM USS-128</td>
<td>X'02' through X'03'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>GS1-128, UCC/EAN 128</td>
<td>X'04'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Intelligent Mail Container Barcode</td>
<td>X'05'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'16'</td>
<td>EAN Two-Digit Supplemental</td>
<td>X'00'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'01'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'17'</td>
<td>EAN Five-Digit Supplemental</td>
<td>X'00'</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'01'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'18'</td>
<td>POSTNET (deprecated) and PLANET (deprecated)</td>
<td>X'00' through X'04'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'1A'</td>
<td>RM4SSC and Dutch KIX</td>
<td>X'00' and X'01'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'1B'</td>
<td>Japan Postal Bar Code</td>
<td>X'00' and X'01'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'1C'</td>
<td>Data Matrix, GS1 DataMatrix (2D bar code)</td>
<td>X'00'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'1D'</td>
<td>MaxiCode (2D bar code)</td>
<td>X'00'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'1E'</td>
<td>PDF417 (2D bar code)</td>
<td>X'00' and X'01'</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Table 11  Modifier Values by Bar Code Type (cont’d.)

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Modifier Value (byte 13)</th>
<th>In BCD1 Subset?</th>
<th>In BCD2 Subset?</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'1F'</td>
<td>Australia Post Bar Code</td>
<td>X'01' through X'08'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'20'</td>
<td>QR Code (2D bar code)</td>
<td>X'02'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'21'</td>
<td>Code 93</td>
<td>X'00'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'22'</td>
<td>Intelligent Mail Barcode</td>
<td>X'00' through X'03'</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>X'23'</td>
<td>Royal Mail RED TAG (deprecated)</td>
<td>X'00'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'24'</td>
<td>GS1 DataBar</td>
<td>X'00' through X'04'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'25'</td>
<td>Royal Mail Mailmark</td>
<td>X'00' and X'01'</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Retired Bar Code Modifier Values

<table>
<thead>
<tr>
<th>Type</th>
<th>Retired item 7</th>
<th>Modifier Value (byte 13)</th>
<th>In BCD1 Subset?</th>
<th>In BCD2 Subset?</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'04'</td>
<td>Retired item 10</td>
<td>X'00'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'0F'</td>
<td>Retired item 11</td>
<td>X'00'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'10'</td>
<td>Retired item 12</td>
<td>X'01' and X'02'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'11'</td>
<td>Retired item 20</td>
<td>X'01'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'12'</td>
<td>Retired item 13</td>
<td>X'01' and X'02'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'13'</td>
<td>Retired item 14</td>
<td>X'01' through X'03'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'14'</td>
<td>Retired item 15</td>
<td>X'00'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'15'</td>
<td>Retired item 16</td>
<td>X'01' and X'02'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'16'</td>
<td>Retired item 17</td>
<td>X'02' through X'03'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'17'</td>
<td>Retired item 18</td>
<td>X'02' through X'03'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'19'</td>
<td>Retired item 19</td>
<td>X'00' through X'03'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'EC'</td>
<td>Retired item 22</td>
<td>X'02'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'ED'</td>
<td>Retired item 23</td>
<td>X'00'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'EE'</td>
<td>Retired item 24</td>
<td>X'00'</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>X'EF'</td>
<td>Retired item 25</td>
<td>X'00' and X'01'</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Refer to "Bar Code Type and Modifier Descriptions" on page 47 for a detailed description of each bar code type and modifier combination.
Byte 14  Local ID

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 28 on page 118. The location of the HRI is specified and varies depending on the symbology selected.

For bar code types that do not allow HRI information, this field is ignored; these are: Data Matrix, Japan Postal Bar Code, MaxiCode, PDF417, POSTNET (deprecated), QR Code, RM4SCC, Royal Mail RED TAG (deprecated), and Royal Mail Mailmark.

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.

Notes:

1. 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.

2. Not all printers can determine the type style or code page of a coded font from the IPDS LFC, LF, LFI, LSS, LCPC, LCP, or LFCSC commands.

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 (note 4 on page 40 describes another way to specify color). Valid values for specifying color include the OCA standard color values (X'0000'–X'0010' and X'FF00'–X'FF08') shown in Table 12 and the special value X'FFFF' that 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.

The specified color value is applied to foreground areas of the bar code presentation space. Foreground areas consist of the following:

- Bars and 2D modules
- Stroked and filled portion of HRI characters

All other areas of the bar code presentation space are background.

Table 12. Standard OCA Color-Value Table

<table>
<thead>
<tr>
<th>Value</th>
<th>Color</th>
<th>Red (R)</th>
<th>Green (G)</th>
<th>Blue (B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'0000' or X'FF00'</td>
<td>Device default</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'0001' or X'FF01'</td>
<td>Blue</td>
<td>0</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>X'0002' or X'FF02'</td>
<td>Red</td>
<td>255</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>X'0003' or X'FF03'</td>
<td>Pink/magenta</td>
<td>255</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>X'0004' or X'FF04'</td>
<td>Green</td>
<td>0</td>
<td>255</td>
<td>0</td>
</tr>
<tr>
<td>X'0005' or X'FF05'</td>
<td>Turquoise/cyan</td>
<td>0</td>
<td>255</td>
<td>255</td>
</tr>
<tr>
<td>X'0006' or X'FF06'</td>
<td>Yellow</td>
<td>255</td>
<td>255</td>
<td>0</td>
</tr>
<tr>
<td>X'0007'</td>
<td>White; see note 1</td>
<td>255</td>
<td>255</td>
<td>255</td>
</tr>
<tr>
<td>X'0008'</td>
<td>Black; see note 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>X'0009'</td>
<td>Dark blue</td>
<td>0</td>
<td>0</td>
<td>170</td>
</tr>
<tr>
<td>X'000A'</td>
<td>Orange</td>
<td>255</td>
<td>128</td>
<td>0</td>
</tr>
<tr>
<td>X'000B'</td>
<td>Purple</td>
<td>170</td>
<td>0</td>
<td>170</td>
</tr>
<tr>
<td>X'000C'</td>
<td>Dark green</td>
<td>0</td>
<td>146</td>
<td>0</td>
</tr>
<tr>
<td>X'000D'</td>
<td>Dark turquoise</td>
<td>0</td>
<td>146</td>
<td>170</td>
</tr>
<tr>
<td>X'000E'</td>
<td>Mustard</td>
<td>196</td>
<td>160</td>
<td>32</td>
</tr>
<tr>
<td>X'000F'</td>
<td>Gray</td>
<td>131</td>
<td>131</td>
<td>131</td>
</tr>
<tr>
<td>X'0010'</td>
<td>Brown</td>
<td>144</td>
<td>48</td>
<td>0</td>
</tr>
<tr>
<td>X'FF07'</td>
<td>Device default</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'FF08'</td>
<td>Color of medium; also known as reset color</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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. It is recommended that OCA Black (X'0008') be rendered as C=M=Y= X'00' and K = X'FF'.
3. 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.

4. In some environments, such as AFP Line Data, IPDS, and MO:DCA environments, colors for the bar code symbol and HRI (using an RGB, CMYK, highlight, or CIELAB color value) can be specified with a Color Specification (X’4E’) triplet. In this case, the Color Specification triplet overrides the color value specified in BSD bytes 15-16. Refer to Appendix C, “IPDS Environment”, on page 145 and Appendix B, “MO:DCA Environment”, on page 143 for more information about color specification in these environments.

**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; refer to Table 14 on page 45 for a list of recommended default values. 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.

**Note:** Most BCOCA implementations support a limited module-width range because device resolution does not allow very small symbols to be accurately produced. The limitations are symbology specific and are commonly in the range 9–36 mils for UPC and EAN symbols and 7–254 mils for most other symbologies; refer to your product documentation for specific ranges supported.

For bar code types that explicitly specify the module width, this field is ignored. Bar code types that explicitly specify the module width are: Australia Post Bar Code, Dutch KIX, Intelligent Mail Barcode, MaxiCode, PLANET (deprecated), POSTNET (deprecated), RM4SCC, Royal Mail RED TAG (deprecated), and Royal Mail Mailmark.

Some bar code types explicitly specify the module width, but allow for a tolerance range in creating the symbol. Some BCOCA receivers can produce either an optimal-size symbol or a small-size symbol for these fixed-size bar codes. This is called “small-symbol support” and is controlled by the value of the module-width parameter, as follows:

**Optimal symbol**

Specify X’FF’ to produce an optimal size symbol. This value is recommended.

**Small symbol**

Specify any value in the range X’01’ – X’FE’ to produce the smallest symbol that meets the symbology tolerances. Because this symbol is at the lower boundary of the symbology-defined tolerance range, external conditions (such as printer contrast setting, toner consistency, and paper absorbency) might cause this symbol to not scan properly.

Note that BCOCA receivers that do not provide “small-symbol support” simply ignore the module-width value (with one exception) and produce an optimal size symbol. The exception is that both options (optimal and small) are supported for Intelligent Mail Barcodes.

The following table describes this option for the fixed-size symbologies.

**Note:** The table provides theoretical sizes. Presentation devices must map the module width specification (or recommendation) to an integer number of device pels. This mapping yields an approximation of the user request and can cause the actual width and height
of a bar code symbol to be slightly different at different device resolutions. Refer to the symbology specification for bar code types that list multiple widths.

**Table 13. Sizing Targets for Fixed-Size Bar Code Types**

<table>
<thead>
<tr>
<th>Bar Code Type</th>
<th>Optimal-Symbol Size</th>
<th>Small-Symbol Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia Post Bar Code</td>
<td>Symbol width = 39.60 mm or 55.85 mm or 72.15 mm</td>
<td>(only with small-symbol support)</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 5.00 mm</td>
<td>Symbol width = 37.0 mm or 52.2 mm or 67.5 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol height = 4.2 mm</td>
</tr>
<tr>
<td>MaxiCode</td>
<td>Symbol width = 25.5 mm</td>
<td>(only with small-symbol support)</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 24.38 mm</td>
<td>Symbol width = 24.03 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol height = 22.98 mm</td>
</tr>
<tr>
<td>POSTNET (deprecated)</td>
<td>Symbol width = 1.429 in or 2.338 in or 2.793 in</td>
<td>(only with small-symbol support)</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 0.125 in</td>
<td>Symbol width = 1.307 in or 2.140 in or 2.557 in</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol height = 0.115 in</td>
</tr>
<tr>
<td>PLANET (deprecated)</td>
<td>Symbol width = 2.793 in or 3.247 in</td>
<td>(only with small-symbol support)</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 0.125 in</td>
<td>Symbol width = 2.557 in or 2.973 in</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol height = 0.115 in</td>
</tr>
<tr>
<td>RM4SCC (for a 5 character symbol)</td>
<td>Symbol width = 38.61 mm</td>
<td>(only with small-symbol support)</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 5.03 mm</td>
<td>Symbol width = 35.31 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol height = 4.22 mm</td>
</tr>
<tr>
<td>Dutch KIX (for an 8 character symbol)</td>
<td>Symbol width = 36.30 mm</td>
<td>(only with small-symbol support)</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 5.03 mm</td>
<td>Symbol width = 33.19 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol height = 4.22 mm</td>
</tr>
<tr>
<td>Intelligent Mail Barcode</td>
<td>Symbol width = 2.95 in</td>
<td>Symbol width = 2.68 in</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 0.145 in</td>
<td>Symbol height = 0.125 in</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Some IPDS printers used the original USPS symbology specification that defined the smallest symbol size as 2.58 inches wide and 0.160 inches high. The USPS specification (Revision B) was changed in 2006 to allow the height of the smallest symbol to be closer to the height of a POSTNET (deprecated) symbol (yielding a smallest symbol size of 2.68 inches wide and 0.134 inches high). In 2007, the specification (Revision D) was changed again to allow the smallest symbol to be 0.125 inches high.</td>
</tr>
<tr>
<td>Royal Mail RED TAG (deprecated)</td>
<td>Symbol width = 56.32 mm</td>
<td>(only with small-symbol support)</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 5.03 mm</td>
<td>Symbol width = 54 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol height = 4.22 mm</td>
</tr>
<tr>
<td>Royal Mail Mailmark</td>
<td>Symbol width (Bar code C) = 79.08 mm</td>
<td>(only with small-symbol support)</td>
</tr>
<tr>
<td></td>
<td>Symbol width (Bar code L) = 93.45 mm</td>
<td>Symbol width (Bar code C) = 69.85 mm</td>
</tr>
<tr>
<td></td>
<td>Symbol height = 5.10 mm</td>
<td>Symbol width (Bar code L) = 82.55 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol height = 4.22 mm</td>
</tr>
</tbody>
</table>

The following equations can be used to convert between L-units, mils, and millimeters, where \( \times \) is the symbol for multiplication and \( \div \) is the symbol for division:

1. Inches \( \times \) (units per unit base) = L-units, also L-units \( \div \) (units per unit base) = inches
For example, when units per unit base is 1440ths, Inches $\times 1440 = \text{L-units}$

2. Inches $\times 1000 = \text{mils}$, also mils / 1000 = inches

3. Inches $\times 25.4 = \text{mm}$, also mm / 25.4 = inches

From (1), (2), and (3) above, using units per unit base of 1440:

mils $\times 1.44 = \text{L-units}$ and mm $\times 1440 / 25.4 = \text{L-units}$

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-multiplierv 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 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 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 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 main-symbol height)

Notes:

1. 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.

2. For Interleaved 2-of-5 symbols, the total height does not include the width of horizontal Bearer Bars placed above and below the symbol.

3. 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.

A value of X'FFFF' indicates the default element height of the presentation device is to be used; refer to Table 14 on page 45 for a list of recommended default values. For bar code types that explicitly specify the element height, this field is ignored; these are: Australia Post Bar Code, Data Matrix, Intelligent Mail Barcode, Japan Postal Bar Code, MaxiCode, POSTNET (deprecated), QR Code, RM4SCC, Royal Mail RED TAG (deprecated), and Royal Mail Mailmark. 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.

The height of GS1 DataBar symbols depends on the version of the symbol. Exception condition EC-0805 exists if the element height and height multiplier values specified are invalid for the modifier selected. Rules for GS1 DataBar symbol heights are as follows:

- GS1 DataBar Omnidirectional – The symbol height specified must be greater than or equal to 33 times the module width.
• GS1 DataBar Truncated – The symbol height specified must be greater than or equal to 13 times the module width.

• GS1 DataBar Stacked – The symbol height is fixed; the element height and height multiplier parameters are ignored.

• GS1 DataBar Stacked Omnidirectional – The row height specified must be greater than or equal to 33 times the module width; the symbol height includes both rows plus the height of the three-module-high separator pattern.

• GS1 DataBar Limited – The symbol height specified must be greater than or equal to 10 times the module width.

• GS1 DataBar Expanded – The symbol height specified must be greater than or equal to 34 times the module width.

• GS1 DataBar Expanded Stacked – The symbol height is fixed; the element height and height multiplier parameters are ignored.

**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, this field is ignored; these are: Australia Post Bar Code, Data Matrix, Intelligent Mail Barcode, Japan Postal Bar Code, MaxiCode, POSTNET (deprecated), QR Code, RM4SCC, Royal Mail RED TAG (deprecated), and Royal Mail Mailmark.

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. Refer to the appropriate symbology specification and printer specification to determine if values outside of the normal range (decimal values below 2.00 and above 3.00) are supported for that symbology; if an unsupported (or invalid) WE:NE value is specified, exception condition EC-0900 exists.

The WE:NE parameter is used with the following bar code types:

- X'02' – MSI (modified Plessey code)
- X'0A' – Industrial 2-of-5
- X'0B' – Matrix 2-of-5
- X'0C' – Interleaved 2-of-5, ITF-14, 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'0002' represents a wide-to-narrow ratio of 2 to 1 and 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; refer to Table 14 on page 45 for a list of recommended default values. 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
Bar Code Symbol Descriptor (BSD)

range of 2.25 through 3.00 to 1. The MSI (modified Plessey code) 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 all bar code types. The Australia Post Bar Code, Code 93, Data Matrix, GS1 DataBar, Intelligent Mail Barcode, Japan Postal Bar Code, MaxiCode, PDF417, POSTNET (deprecated), QR Code, RM4SCC, Royal Mail RED TAG (deprecated), and Royal Mail Mailmark 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.
Default Value Recommendations

It is desirable that BCOCA implementations be reasonably consistent so that print jobs appear essentially the same regardless of which printer prints the job and regardless of which transform or display product creates bar code symbols from BCOCA input. The following table provides recommendations for what BCOCA implementations should use when the default module width, element height, or wide-to-narrow ratio is specified. Many BCOCA implementations existed before these recommendations were first published; refer to your printer documentation for the exact default values used by your printer.

Some bar code symbologies explicitly specify the module width or element height; in these cases, the following table lists the module width or element height value defined for the symbology. Refer to the description of module width (byte 17) and element height (bytes 18–19) for a list of the symbologies that explicitly specify these values.

Table 14. Recommended Default Values for Module Width, Element Height, and Wide-to-Narrow Ratio

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Recommend Default Module Width</th>
<th>Recommended Default Element Height</th>
<th>Recommended Default Wide-to-Narrow Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Code 39 (3-of-9 Code), AIM USS-39</td>
<td>13 mils</td>
<td>Greater of 250 mils or 15% of symbol width</td>
<td>2.5</td>
</tr>
<tr>
<td>X'02'</td>
<td>MSI (modified Plessey code)</td>
<td>13 mils</td>
<td>Greater of 300 mils or 15% of symbol width</td>
<td>2.0</td>
</tr>
<tr>
<td>X'03'</td>
<td>UPC/CGPC-Version A</td>
<td>13 mils</td>
<td>1020 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'05'</td>
<td>UPC/CGPC-Version E</td>
<td>13 mils</td>
<td>1020 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'06'</td>
<td>UPC—Two-Digit Supplemental (Periodicals)</td>
<td>13 mils</td>
<td>770 mils (bar height)</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'07'</td>
<td>UPC—Five-Digit Supplemental (Paperbacks)</td>
<td>13 mils</td>
<td>770 mils (bar height)</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'08'</td>
<td>EAN-8 (includes JAN-short)</td>
<td>13 mils</td>
<td>840 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'09'</td>
<td>EAN-13 (includes JAN-standard)</td>
<td>13 mils</td>
<td>1020 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'0A'</td>
<td>Industrial 2-of-5</td>
<td>13 mils</td>
<td>Greater of 250 mils or 15% of symbol width</td>
<td>2.5</td>
</tr>
<tr>
<td>X'0B'</td>
<td>Matrix 2-of-5</td>
<td>13 mils</td>
<td>Greater of 250 mils or 15% of symbol width</td>
<td>2.5</td>
</tr>
<tr>
<td>X'0C'</td>
<td>Interleaved 2-of-5, ITF-14, AIM USS-I 2/5</td>
<td>13 mils²</td>
<td>Greater of 250 mils or 15% of symbol width²</td>
<td>2.5</td>
</tr>
<tr>
<td>X'0D'</td>
<td>Codabar, 2-of-7, AIM USS-Codabar</td>
<td>13 mils</td>
<td>Greater of 250 mils or 15% of symbol width</td>
<td>2.5</td>
</tr>
<tr>
<td>X'11'</td>
<td>Code 128, AIM USS-128 Code 128 modifier X'02'</td>
<td>13 mils</td>
<td>Greater of 250 mils or 15% of symbol width</td>
<td>Not applicable</td>
</tr>
<tr>
<td></td>
<td>UCC/EAN 128 Code 128 modifier X'03'</td>
<td>13 mils</td>
<td>Greater of 250 mils or 15% of symbol width</td>
<td>Not applicable</td>
</tr>
<tr>
<td></td>
<td>UCC/EAN 128 and GS1-128 Code 128 modifier X'04'</td>
<td>13 mils</td>
<td>Greater of 250 mils or 15% of symbol width</td>
<td>Not applicable</td>
</tr>
<tr>
<td></td>
<td>Intelligent Mail Container Barcode Code 128 modifier X'05'</td>
<td>25 mils</td>
<td>925 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'16'</td>
<td>EAN Two-Digit Supplemental</td>
<td>13 mils</td>
<td>840 mils (bar height)</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>
### Table 14  Recommended Default Values for Module Width, Element Height, and Wide-to-Narrow Ratio (cont’d.)

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Recommend Default Module Width</th>
<th>Recommended Default Element Height</th>
<th>Recommended Default Wide-to-Narrow Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'17'</td>
<td>EAN Five-Digit Supplemental</td>
<td>13 mils</td>
<td>840 mils (bar height)</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'18'</td>
<td>POSTNET (deprecated) and PLANET (deprecated)</td>
<td>20 mils with a horizontal pitch of 22 bars/inch</td>
<td>125 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'1A'</td>
<td>RM4SCC and Dutch KIX</td>
<td>20 mils with a horizontal pitch of 22 bars/inch</td>
<td>198 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'1B'</td>
<td>Japan Postal Bar Code</td>
<td>24 mils</td>
<td>6 times module width</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'1C'</td>
<td>Data Matrix, GS1 DataMatrix (2D bar code)</td>
<td>21 mils</td>
<td>21 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'1D'</td>
<td>MaxiCode (2D bar code)</td>
<td>Defined in symbology</td>
<td>Defined in symbology</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'1E'</td>
<td>PDF417 (2D bar code)</td>
<td>14 mils</td>
<td>4 times module width</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'1F'</td>
<td>Australia Post Bar Code</td>
<td>20 mils with a horizontal pitch of 23.5 bars/inch</td>
<td>197 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'20'</td>
<td>QR Code (2D bar code)</td>
<td>12 mils</td>
<td>12 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'21'</td>
<td>Code 93</td>
<td>13 mils</td>
<td>Greater of 250 mils or 15% of symbol width</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'22'</td>
<td>Intelligent Mail Barcode</td>
<td>20 mils with a horizontal pitch of 22 bars/inch</td>
<td>145 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'23'</td>
<td>Royal Mail RED TAG (deprecated)</td>
<td>20 mils with a horizontal pitch of 23 bars/inch</td>
<td>198 mils</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'24'</td>
<td>GS1 DataBar: Omnidirectional (X'00') Truncated (X'01') Stacked (X'02') Stacked - Omnidirectional(X'03') Limited (X'04') Expanded (X'11') Expanded - Stacked (X'12'–X'1B')</td>
<td>10 mils</td>
<td>33 times modwidth 13 times modwidth not applicable 33 times modwidth 10 times modwidth 34 times modwidth not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>X'25'</td>
<td>Royal Mail Mailmark</td>
<td>21 mils with a horizontal pitch of 21.2 bars/inch</td>
<td>201 mils</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

**Notes:**

1. Module width measures the width of the smallest bar in the symbol and, for most bar codes, this is also the size of the smallest space. However, some postal bar codes specify symbol width in terms of bar width and also horizontal pitch. Horizontal pitch measures the number of bars per inch (or bars per 25.4 mm); this typically causes the spaces between bars to be different than the bar width.

2. The module width and element height for ITF-14 symbols is defined by the application specification based on the needs of the application. Therefore, the default values might not be appropriate for all applications of the ITF-14 symbol; refer to GS1 General Specifications for more information.
Bar Code Type and Modifier Descriptions

Each bar code type supports one or more variations that are specified with a modifier value, as follows:

Code 39 (3-of-9 Code), AIM USS-39 (modifier values X'01' and X'02')

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 alphabets, and some special characters. The Code 39 Specification also provides a method of encoding all 128 ASCII characters by using two 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 two 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 two-character combination bar/space pattern appropriately.
MSI (modified Plessey code, modifier values X'01' through X'09')

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 (modifier value X'00')

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.

Note: The UPC-A symbology is controlled by the GS1 standards organization and is described in *GS1 General Specifications*.

UPC/CGPC — Version E (modifier value X'00')

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).

Note: The UPC-E symbology is controlled by the GS1 standards organization and is described in *GS1 General Specifications*. 
UPC—Two-Digit Supplemental

UPC – Two-Digit Supplemental (modifier values X'00' through X'02')

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 UPC Two-Digit 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 (must be 0), 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-odd and left-even parity as determined by the two supplemental digits.

Specify 13 digits of input data.

X'02' The UPC Two-Digit 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-odd and left-even parity as determined by the two digits.

Specify 12 digits of input data.

Note: The UPC Two-Digit Supplemental symbology is controlled by the GS1 standards organization and is described in GS1 General Specifications.
UPC – Five-Digit Supplemental (modifier values X'00' through X'02')

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 UPC Five-Digit 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 (must be 0), 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-odd and left-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 UPC Five-Digit 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-odd and left-even parity. The supplement check digit is not encoded or interpreted.

Specify 15 digits of input data.

Note: The UPC Five-Digit Supplemental symbology is controlled by the GS1 standards organization and is described in GS1 General Specifications.
EAN-8 and EAN-13

EAN-8 (includes JAN-short, modifier value X'00')

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.

Note: The EAN-8 symbology is controlled by the GS1 standards organization and is described in GS1 General Specifications.

EAN-13 (includes JAN-standard, modifier value X'00')

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.

Note: The EAN-13 symbology is controlled by the GS1 standards organization and is described in GS1 General Specifications.
2-of-5 Codes

Industrial 2-of-5 (modifier values X'01' and X'02')

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 (modifier values X'01' and X'02')

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, ITF-14, AIM USS-I 2/5 (modifier values X'01' through X'04')

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.
2-of-5 Codes

X'03' Present the barcode symbol with a generated check digit and with Bearer Bars that completely surround the bar/space pattern.

The purpose of Bearer Bars is to reduce the possibility of misreads or short scans that might occur when a skewed scanning beam enters or exits the barcode symbol through its top or bottom edge. Bearer Bars should be a constant minimum thickness of twice the width of the narrow bar, placed directly against the top, bottom, and sides of the symbol plus quiet zone. The Bearer Bars should completely surround the symbol, including the quiet zones, which are a minimum of 10 times the X dimension.

![ITF-14 Symbol with Surrounding Bearer Bars](image)

X'04' Present the barcode symbol with a generated check digit and with Bearer Bars that are placed at the top and the bottom of the bar/space pattern.

The purpose of Bearer Bars is to reduce the possibility of misreads or short scans that might occur when a skewed scanning beam enters or exits the barcode symbol through its top or bottom edge. Bearer Bars should be a constant minimum thickness of twice the width of the narrow bar, placed directly against the top and bottom of the symbol bars.

![Interleaved 2-of-5 Symbol with Bearer Bars at Top and Bottom](image)

**Note:** ITF-14 is a special case of Interleaved 2-of-5, which encodes 13 input digits and a check digit. The ITF-14 symbology is controlled by the GS1 standards organization and is described in *GS1 General Specifications*. 
Codabar, 2-of-7, AIM USS-Codabar (modifier values X'01' and X'02')

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

Code 128 (modifier values X'02' through X'05')

Code 128 is a general purpose symbology that has been used in several ways. BCOCA architecture uses the following modifiers to support some of these uses:

Modifier X'02’ – AIM USS-128
This is a basic Code 128 symbol that is defined in USS-128 Uniform Symbology Specification published by AIM.

Modifiers X'03' – UCC/EAN 128
This is a variation of the Code 128 symbol that was originally defined in UCC/EAN-128 Application Identifier Standard and the Application Standard for Shipping Container Codes published by the Uniform Code Council and was also defined by the European Article Numbering Association (EAN). A newer description of the UCC/EAN 128 symbology is available in GS1 General Specifications. The GS1 standards group became the successor to the organizations previously known as EAN and UCC. Many BCOCA implementations use the earlier specifications.

Modifier X'04’ – UCC/EAN 128 and GS1-128
This is a variation of the Code 128 symbol identical to modifier 03 except that parentheses are used in the HRI to distinguish each application identifier (ai). A UCC/EAN-128 symbol can use either modifier X'03' or modifier X'04'. GS1-128 symbols use modifier X'04'.

Modifier X'05’ – Intelligent Mail Container Barcode
This is a bar code that is defined in BARCODE, CONTAINER, INTELLIGENT MAIL (USPS-B-3215) published by the United States Postal Service (USPS). The bar code uses a special form of the GS1-128 symbol that is defined in GS1 General Specifications published by GS1.

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' or modifier X'04' 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 and GS1-128 bar codes is in the form:

\[
FNC1, \text{ai, data, [m]}, [FNC1], \text{ai, data, [m]}, [FNC1], \ldots, \text{ai, data, [m]}
\]

Where “FNC1” is the FNC1 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 “FNC1” 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 “FNC1”. 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 “FNC1”, “ai”, “data”, and “m” check digit
3. produce a modulo 103 check digit
4. produce a stop character.

The Intelligent Mail Tray Barcode defined by the United States Postal Service uses the Code 128 bar code symbology.
**Code 128 modifier X’02’ – Code 128 symbol, using original (1986) start-character algorithm**

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, GCSSID = 1454) is used to interpret the bar code symbol data. Generate a check digit and present it with the bar code.

**Note:** Some IPDS printers incorrectly 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 that 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.

IPDS printers should use the original start-character algorithm when modifier X’02’ is specified. Known printers that incorrectly use the UCC/EAN 128 start-character algorithm when modifier X’02’ is specified include: IBM 4312, IBM 4317, IBM 4324, Infoprint® 20, Infoprint 21, Infoprint 32, Infoprint 40, Infoprint 45, Infoprint 70, Infoprint 2070, Infoprint 2085, and Infoprint 2105.
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 FNC1 (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.

The UCC/EAN 128 data is checked for validity and exception condition EC-1200 exists if one or more of the following conditions are encountered:
- FNC1 is not the first data character
- Invalid application identifier (ai) value encountered
- Data for an ai doesn't match the ai definition
- Insufficient (or no) data following an ai
- Too much data for an ai
- Invalid use of FNC1 character

Notes:

1. UCC/EAN 128 is a variation of Code 128 that begins with an FNC1 character, followed by an Application Identifier and the data to be bar encoded. All of these characters (including the FNC1 character) must be supplied within the Bar Code Symbol Data (BSA). UCC/EAN 128 also requires that the symbol begin in subset C. The GS1-128 symbology allows symbols to begin with either subset A, B, or C.

2. For UCC/EAN 128 symbols, the start character, the FNC1 characters, the modulo 103 check digit, and the stop character are not shown in the human readable format.
**Code 128 modifier X'04' – UCC/EAN 128 and GS1-128 symbols, with parentheses in the HRI**

Generate a Code 128 symbol in the same manner as for modifier X'03', but use parentheses in the HRI to distinguish each application identifier (ai). The printer inserts the parentheses in the printed HRI when modifier X'04' is specified; these parentheses are not part of the input data.

**Note:** The GS1-128 symbology is controlled by the GS1 standards organization and is described in GS1 General Specifications.
The Intelligent Mail Container Barcode symbology is defined and used by the United States Postal Service (USPS) for the Full Service category of automation discounts. The bar code uses a special form of the GS1-128 (also known as UCC/EAN 128) symbology for printing on mailer-generated pallet labels to uniquely identify pallets and similar containers and to identify the mail owner; a unique serial number can also be provided for each container.

The printer will generate a GS1-128 symbol as described in the USPS symbology specification (BARCODE, CONTAINER INTELLIGENT MAIL); the GS1-128 Specification is used to produce the bar code symbol. The Code 128 code page is used to interpret the bar code symbol data (CPGID = 1303, GCSPID = 1454; refer to Figure 14 on page 129). The printer will also produce an appropriate USPS Banner (USPS SCAN REQUIRED) and Identification Bars above and below the symbol. If requested, HRI will be printed below the symbol using two blanks as separators between each field of the HRI.

The Intelligent Mail Container Barcode symbology allows for a variety of symbol sizes. The module width must be between 23 mils and 27 mils and the height must be between 0.75 inches and 1.1 inches. A symbol width between 6.25 inches and 7.25 inches is recommended.

The input data for the bar code is alphanumeric and consists of 22 characters as shown in Table 15. The serial number field must be padded on the left with either leading zeros (code point X’F0’) or leading dashes (code point X’60’); leading zeros are recommended. The BCoca symbol data is checked for validity and exception condition EC-1203 exists if the data is invalid or insufficient.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Source</th>
<th>Field Size and Data Type</th>
<th>Field Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function 1 Symbol Character</td>
<td>USPS assigned</td>
<td>1 byte</td>
<td>FNC1 (X’8F’)</td>
</tr>
<tr>
<td>Application Identifier</td>
<td>USPS assigned</td>
<td>2 bytes (numeric)</td>
<td>99</td>
</tr>
<tr>
<td>Type Indicator</td>
<td>USPS assigned</td>
<td>1 byte (alphabetic)</td>
<td>M</td>
</tr>
<tr>
<td>Mailer ID</td>
<td>USPS assigned</td>
<td>either 6 bytes or 9 bytes (numeric)</td>
<td>Six-byte Mailer IDs are in the range 0000000–899999 Nine-byte Mailer IDs are in the range 900000000–999999999</td>
</tr>
<tr>
<td>Serial Number</td>
<td>Mailer assigned</td>
<td>either 12 bytes or 9 bytes (alphanumeric)</td>
<td>Any alphanumeric value When the Mailer ID is 6 bytes, the Serial Number is 12 bytes When the Mailer ID is 9 bytes, the Serial Number is 9 bytes</td>
</tr>
</tbody>
</table>
The user must provide sufficient white space around the bar code for quiet zones (the printer does not provide the quiet zones). A quiet zone of at least 0.125 inches is required above and below the bar code. A quiet zone of at least 10 times the module width is required to the left and right of the bar code.

The origin of the bar code symbol is defined to be the top-left corner of an imaginary rectangle of minimum size that bounds the bar and space pattern. Since the HRI, USPS Banner, Identification Bars, and quiet zone are outside of the imaginary rectangle, it is important to make sure that the symbol is positioned to allow for these items. If any part of the symbol, HRI, USPS Banner, or Identification Bars fall outside the bar code presentation space, exception ID EC-1100 exists.
EAN Two-Digit Supplemental

EAN Two-Digit Supplemental (modifier values X'00' and X'01')

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-odd and left-even parity as determined by the two supplemental digits chosen from number sets A and B.

Specify 14 digits of input data.

Note: The EAN Two-Digit Supplemental symbology is controlled by the GS1 standards organization and is described in GS1 General Specifications.
EAN Five-Digit Supplemental (modifier values X'00' and X'01')

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: The EAN Five-Digit Supplemental symbology is controlled by the GS1 standards organization and is described in GS1 General Specifications.
POSTNET and PLANET (both deprecated, modifier values X'00' through X'04')

<table>
<thead>
<tr>
<th>US POSTNET</th>
<th>PLANE Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zip+4</td>
<td>(encoding 12345+6789)</td>
</tr>
</tbody>
</table>

**Note:** The POSTNET and PLANET symbologies have been retired by the United States Postal Service and have also been deprecated in the BCOCA architecture. For a description of the replacement, refer to the “Intelligent Mail Barcode (modifier values X'00' through X'03')” on page 75.

For all POSTNET modifiers that follow, the BSA HRI flag field and the BSD 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.

Some BCOCA implementations use the module width parameter to specify one of two symbol sizes (small or optimal); refer to the description of module width on page 40 for details. This function is called small-symbol support; printers that do not provide small-symbol support ignore the module width field.

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 mail piece. 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 and Dutch KIX (modifier values X'00' and X'01')

This is 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 an 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 an RM4SCC bar code symbol with no start bar, no checksum character, and no stop bar.
This is 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, check digit, 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 checksum character, 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 nnnnnnnn; 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 that 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 barcode 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 that can be numeric, hyphen, or control characters (CC1 through CC8). The following table lists the valid code points for modifier X'01'.
### Table 16. Valid Code Points for Direct Input to a Japan Postal Bar Code

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

**Implementation Note:**

These code points are EBCDIC-based to match early Japan Postal Bar Code implementations that used fonts instead of BCOCA; there is no known requirement for ASCII-based code points.
Data Matrix and GS1 DataMatrix (modifier value X'00')

This is 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 that 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 (that is a backslash in the default character encodation); alternatively, the escape-sequence handling flag (see page 95) 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 95) 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.

Note: The Data Matrix bar code is used for many applications and some of these applications have specific rules that must be followed when specifying the parameters and data for the bar code object. For example, some applications require a particular encodation scheme; therefore, an IPDS printer used to print the symbol must support both Data Matrix and the encodation scheme option (STM property pair X'1303'). Examples of Data Matrix applications with special rules include the following:

- The GS1 DataMatrix symbology is controlled by the GS1 standards organization and is described in the *GS1 General Specifications*.

- The Royal Mail Complex Mail Data Marks (CMDM) symbology is controlled by Royal Mail and is described in the *Royal Mail Mailmark Definition Document*. CMDM symbols use the C40 encodation scheme.
This is 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 encoding 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 that 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 (that is a backslash in the default character encodation); alternatively, the escape-sequence handling flag (see page 100) 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 100) 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.

**Note:** Care should be taken when using the End-of-Transmission (EOT) character; many MaxiCode examples show EOT as the last character of the data. It has been reported that for MaxiCode symbols that will be scanned by the United Parcel Service (the originator of MaxiCode), the EOT must not be followed by additional characters. However, the MaxiCode symbology specification does not contain any special rules for handling EOT characters or data found after an EOT. Because of this inconsistency, how data after an EOT is handled is device specific; some BCOCA receivers encode all of the data, some ignore data after EOT, and some provide a device-specific way to inform the BCOCA receiver how to handle data after EOT.
PDF417 (modifier values X'00' and X'01')

This is 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 (that is a backslash in the default character encodation); alternatively, the escape-sequence handling flag (see page 107) 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 that 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 PDF417 symbology specification.

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 PDF417 symbology specification. 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 PDF417 symbology specification. 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 105) 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 (modifier values X'01' through X'08')

Australia Post Bar Code
Customer Barcode 2 using Table C
(encoding 56439111ABA9)

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

Table 17. Australia Post Modifier Values

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Type of Bar Code</th>
<th>Valid Bar Code Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Standard Customer Barcode (format code = 11)</td>
<td>An 8 digit number representing the Sorting Code</td>
</tr>
<tr>
<td>X'02'</td>
<td>Customer Barcode 2 using Table N (format code = 59)</td>
<td>An 8 digit number representing the Sorting Code followed by up to 8 numeric digits representing the Customer Information</td>
</tr>
<tr>
<td>X'03'</td>
<td>Customer Barcode 2 using Table C (format code = 59)</td>
<td>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</td>
</tr>
<tr>
<td>X'04'</td>
<td>Customer Barcode 2 using proprietary encoding (format code = 59)</td>
<td>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</td>
</tr>
<tr>
<td>X'05'</td>
<td>Customer Barcode 3 using Table N (format code = 62)</td>
<td>An 8 digit number representing the Sorting Code followed by up to 15 numeric digits representing the Customer Information</td>
</tr>
<tr>
<td>X'06'</td>
<td>Customer Barcode 3 using Table C (format code = 62)</td>
<td>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</td>
</tr>
<tr>
<td>X'07'</td>
<td>Customer Barcode 3 using proprietary encoding (format code = 62)</td>
<td>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</td>
</tr>
<tr>
<td>X'08'</td>
<td>Reply Paid Barcode (format code = 45)</td>
<td>An 8 digit number representing the Sorting Code</td>
</tr>
</tbody>
</table>

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 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 and should be printed above the symbol. The format control code, Delivery Point Identifier, and customer information field (if any) appears in the HRI, but the ECC does not.
QR Code (modifier value X'02')

This is 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 112) can be used in the following manner:

• When all of the input data for the bar code is single-byte EBCDIC using one of the supported code pages (500, 290, or 1027), set the EBCDIC-to-ASCII translation flag to B'1' and select the correct code page in the conversion parameter.

• When all of the input data for the bar code is mixed-byte EBCDIC AFP Line Data using SO and SI controls (SOSI data), set the EBCDIC-to-ASCII translation flag to B'1' and select the desired conversion value in the conversion parameter.

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 those previously mentioned (such as, in an EBCDIC code page not supported by the EBCDIC-to-ASCII translation flag), the bar code data must begin in the default encodation, the EBCDIC-to-ASCII translation flag must be set to B'0', and the ECI protocol must be used to switch into the other encodation(s).

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.
This is 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 alphabets, 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.
Intelligent Mail Barcode (modifier values X'00' through X'03')

The Intelligent Mail Barcode symbology limits the symbol size; therefore BSD element height, height multiplier, and wide-to-narrow ratio fields are not applicable to this symbology and are ignored by BCOCA receivers. The module width field allows for two symbol sizes (small and optimal). The small symbol is approximately 2.68 inches wide and 0.125 inches high. The optimal symbol is approximately 2.95 inches wide and 0.145 inches high.

The input data is all numeric and consists of 5 data fields. The first four fields are essentially fixed length and the 5th field can have one of four lengths; the bar code modifier is used to specify the length of the 5th field. The total length of the input data can be 20, 25, 29, or 31 digits that is defined as follows:

- Barcode ID (2 digits) – assigned by USPS, the 2nd digit must be 0–4; thus, the valid values are: 00–04, 10–14, 20–24, 30–34, 40–44, 50–54, 60–64, 70–74, 80–84, and 90–94
- Service Type ID (3 digits) – assigned by USPS; valid values are 000–999
- Mailer ID fields; 15 digits in the range 000000000000000–999999999999999
  – Mailer ID (6 or 9 digits) – assigned by USPS
  – Sequence or serial number (9 or 6 digits) – assigned by the mailer
- Routing ZIP Code (0, 5, 9, or 11 digits) – refer to the modifier for valid values; also called Delivery Point ZIP Code

Intelligent Mail Barcode modifier values are defined as follows:

X'00' Present an Intelligent Mail Barcode symbol with no Routing ZIP Code. The input data for this bar code symbol must be 20 numeric digits.

X'01' Present an Intelligent Mail Barcode symbol with a 5-digit Routing ZIP Code. The input data for this bar code symbol must be 25 numeric digits; the valid values for the Routing ZIP Code are 00000–99999.

X'02' Present an Intelligent Mail Barcode symbol with a 9-digit Routing ZIP Code. The input data for this bar code symbol must be 29 numeric digits; the valid values for the Routing ZIP Code are 000000000–999999999.

X'03' Present an Intelligent Mail Barcode symbol with an 11-digit Routing ZIP Code. The input data for this bar code symbol must be 31 numeric digits; the valid values for the Routing ZIP Code are 00000000000–99999999999.

Human-Readable Interpretation (HRI) can be printed with an Intelligent Mail Barcode symbol, but HRI is not used with all types of special services. Refer to *Introducing 4-state Customer Barcode* for a description of when HRI is appropriate.

There must be a quiet zone surrounding the symbol (all four sides) that is at least 0.04 inches above and below and at least 0.125 inches on both sides of the symbol.

---

1. The United States Postal Service (USPS) developed this symbology for use in the USPS mail stream and has named it the "Intelligent Mail Barcode". Originally, BCOCA architecture used the name “USPS Four-State bar code” for this symbology. The bar code is also known as the “OneCode SOLUTION Barcode” and the “4-state Customer Barcode” and has been abbreviated in several ways: OneCode (4CB), OneCode (4-CB), 4CB, or 4-CB.
Royal Mail RED TAG (deprecated), modifier value X'00'

Note: The RED TAG symbology has been retired by Royal Mail and has also been deprecated in the BCOCA architecture. For a description of the replacement, refer to “Royal Mail Mailmark” on page 83.

The RED TAG bar code symbology is defined and used by Royal Mail Group Ltd. for intelligent mail tracking and reporting. The RED TAG bar code is a four-state symbol with exactly 51 bars that includes a RED TAG indicator printed at each end of the symbol.

The Royal Mail RED TAG symbology limits the symbol size; therefore BSD element height, height multiplier, and wide-to-narrow ratio fields are not applicable to this symbology and are ignored by BCOCA receivers. The module width field allows for two symbol sizes (small and optimal); the small symbol is approximately 2.13 inches wide and the optimal symbol is approximately 2.22 inches wide.

The input data for the bar code is all numeric and consists of the fields shown in Table 18 (in the specified order). The value ranges are those defined within the first version of the RED TAG symbology specification, but to allow for future expansion, BCOCA allows a larger range for each field. Values outside of the “RED TAG Recommended Range” should not be used by the user. The RED TAG data is checked for validity (within the BCOCA range) and exception condition EC-1202 exists if the data is invalid or insufficient. There must be exactly 21 numeric digits; if needed, each field is padded on the left with zeroes to fill the field. For example, “012345672250513234567” would be specified for the following RED TAG input fields:

- Account ID = 12345
- Product ID = 67
- Class = 2
- Day = 25
- Month = 5
- Consignment ID = 13
- Item Unique ID = 234567

Table 18. Royal Mail RED TAG (deprecated) Data Field Ranges

<table>
<thead>
<tr>
<th>External Field Name</th>
<th>Source</th>
<th>Field Size</th>
<th>BCOCA Range</th>
<th>RED TAG Recommended Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account ID</td>
<td>Royal Mail</td>
<td>6 bytes</td>
<td>1–213,868</td>
<td>1–200,000</td>
</tr>
<tr>
<td>Product ID</td>
<td>Royal Mail</td>
<td>2 bytes</td>
<td>0–99</td>
<td>1–99</td>
</tr>
<tr>
<td>Class</td>
<td>Mailer</td>
<td>1 byte</td>
<td>0–3</td>
<td>1–3</td>
</tr>
<tr>
<td>Day</td>
<td>Mailer</td>
<td>2 bytes</td>
<td>1–31</td>
<td>1–31</td>
</tr>
<tr>
<td>Month</td>
<td>Mailer</td>
<td>2 bytes</td>
<td>1–12</td>
<td>1–12</td>
</tr>
<tr>
<td>Consignment ID</td>
<td>Mailer</td>
<td>2 bytes</td>
<td>0–49</td>
<td>1–49</td>
</tr>
<tr>
<td>Item Unique ID</td>
<td>Mailer</td>
<td>6 bytes</td>
<td>0–249,999</td>
<td>1–249,999</td>
</tr>
</tbody>
</table>
The Royal Mail RED TAG bar code type only uses one modifier value:

X'00'  Present a Royal Mail RED TAG bar code symbol with a RED TAG indicator printed at each end of the symbol. The RED TAG indicator is a capital “O” printed in Arial 20 point bold type.

Human-Readable Interpretation (HRI) is not used with the Royal Mail RED TAG symbol.

There must be a 5 mm quiet zone surrounding the symbol (all four sides); the RED TAG indicator is outside of the quiet zone.

The origin of the bar code symbol is defined to be the top-left corner of an imaginary rectangle of minimum size that bounds the bar and space pattern. Since the RED TAG indicator and the quiet zone are outside of the imaginary rectangle, it is important to make sure that the symbol is positioned at least 10 mm from the left edge of the bar code presentation space. If any part of the symbol or RED TAG indicator falls outside the bar code presentation space, exception ID EC-1100 exists.
GS1 DataBar

GS1 DataBar is a family of bar codes that is designed for items for which traditional linear bar codes are too large or are inconveniently shaped. The GS1 DataBar family has seven versions (selected with modifiers X'00' – X'04' and X'11' – X'1B'):

- The first group requires 14 numeric digits as input. There are four versions in this group that have identical encodation rules and structure, but different shapes:
  - GS1 DataBar Omnidirectional (modifier X'00')
  - GS1 DataBar Truncated (modifier X'01')
  - GS1 DataBar Stacked (modifier X'02')
  - GS1 DataBar Stacked Omnidirectional (modifier X'03')

- The second group, called GS1 DataBar Limited (modifier X'04'), is structurally different, has different encodation rules, and requires 14 numeric digits as input (the first digit must be 0 or 1).

- The third group, called GS1 DataBar Expanded, has yet another symbology structure and different encodation rules. The format of the input data for GS1 DataBar Expanded is exactly the same as the input data for a UCC/EAN 128 bar code. There are two versions of GS1 DataBar Expanded:
  - GS1 DataBar Expanded (modifier X'11')
  - GS1 DataBar Expanded Stacked (modifiers X'12' – X'1B')

The GS1 DataBar Omnidirectional, Stacked Omnidirectional, Expanded, and Expanded Stacked symbols can be read in segments by omnidirectional scanners.

The height of the symbol is different for each version (modifier value). Because the first element of each bar code symbol is a space, no quiet zone is needed for this bar code.

Human-Readable Interpretation (HRI) can be printed below an GS1 DataBar symbol. The content of the HRI depends on the version of the symbol:

- For modifiers X'00' – X'04', the HRI consists of implied application ID 01 in parentheses followed by the 14 digit input data. The implied application ID is not part of the input data, nor is it included within the symbol. An example of HRI for GS1 DataBar symbols is shown in each modifier description.

- For modifiers X'11' – X'1B', the HRI consists of the input data with the application IDs surrounded by parentheses and the FNC1 characters suppressed.

**Modifier X'00'**

Present a GS1 DataBar Omnidirectional bar code symbol. The height of the symbol must be greater than or equal to 33 times the module width.

The input data for this bar code symbol is 14 numeric digits that conform to application identifier 01. The bar code receiver will compact the data, create guard patterns, create data-character patterns, calculate a checksum, create finder patterns, and generate a GS1 DataBar Omnidirectional bar code symbol.
Modifier X'01'

Present a GS1 DataBar Truncated bar code symbol. This is the same as the standard Omnidirectional symbol except that its height is reduced to a minimum of 13 times the module width.

The input data for this bar code symbol is 14 numeric digits that conform to application identifier 01. The bar code receiver will compact the data, create guard patterns, create data-character patterns, calculate a checksum, create finder patterns, and generate a GS1 DataBar Truncated bar code symbol.

Modifier X'02'

Present a GS1 DataBar Stacked bar code symbol. This is the same as the standard Omnidirectional symbol except that its height is fixed and it is presented in two stacked rows with a separator pattern between the rows.

The input data for this bar code symbol is 14 numeric digits that conform to application identifier 01. The bar code receiver will compact the data, create guard patterns, create data-character patterns, calculate a checksum, create finder patterns, and generate a GS1 DataBar Stacked bar code symbol.

Modifier X'03'

Present a GS1 DataBar Stacked Omnidirectional bar code symbol. This is the same as the standard Omnidirectional symbol except that it is presented in two stacked rows with a separator pattern between the rows. Like the Omnidirectional symbol, the height of each of the two rows must be greater than or equal to 33 times the module width.

The input data for this bar code symbol is 14 numeric digits that conform to application identifier 01. The bar code receiver will compact the data, create guard patterns, create data-character patterns, calculate a checksum, create finder patterns, and generate a GS1 DataBar Stacked Omnidirectional bar code symbol.
GS1 DataBar

Modifier X'04'

Present a GS1 DataBar Limited bar code symbol. The height of the symbol must be greater than or equal to 10 times the module width.

The input data for this bar code symbol is 14 numeric digits that conform to application identifier 01; however, the first digit must be 0 or 1. The bar code receiver will compact the data, create guard patterns, create data-character patterns, calculate a checksum, create a finder pattern, and generate a GS1 DataBar Limited bar code symbol.

Modifier X'11'

Present a GS1 DataBar Expanded bar code symbol. The height of the symbol must be greater than or equal to 34 times the module width.

The format of the input data for this bar code symbol (up to 74 numeric digits or up to 41 alphabetic characters) is similar to that of a UCC/EAN 128 bar code; refer to the description on page 56 for a description of UCC/EAN 128. The difference is that UCC/EAN 128 symbols must begin with an FNC1 character. The data for GS1 DataBar Expanded bar code is of the form:

\[ ai, \text{data}, [m], [FNC1], ai, \text{data}, [m], [FNC1], \ldots, ai, \text{data}, [m] \]

The GS1 DataBar Expanded data is checked for validity and exception condition EC-1200 exists if one or more of the following conditions are encountered:

- Invalid application identifier (ai) value encountered
- Data for an ai doesn't match the ai definition
- Insufficient (or no) data following an ai
- Too much data for an ai
- Invalid use of FNC1 character

**Note:** Because the data for an Expanded symbol is similar to the data for a UCC/EAN 128 symbol, BCOCA receivers will tolerate FNC1 characters that precede the first ai by ignoring them.

The bar code receiver will compact the data, pad the binary data with the B'00100' padding string until sufficient symbol characters are built, create guard patterns, create data-character patterns, calculate a check character, create finder patterns, and generate a GS1 DataBar Expanded bar code symbol.
Modifiers X'12' – X'1B'

Present a GS1 DataBar Expanded Stacked bar code symbol. This is the same as the standard GS1 DataBar Expanded symbol except that it is presented in stacked rows with a separator pattern between the rows. Expanded Stacked symbols are typically narrower than the equivalent Expanded version because they allow the bar code to trade vertical space for horizontal space. The specific modifier value provides control over symbol width by identifying a requested number of symbol characters per row as shown in the following table:

<table>
<thead>
<tr>
<th>Modifier Value</th>
<th>Requested Number of Symbol Characters per Row</th>
<th>Width of Symbol in Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'12'</td>
<td>2 per row</td>
<td>53 modules</td>
</tr>
<tr>
<td>X'13'</td>
<td>4 per row</td>
<td>102 modules</td>
</tr>
<tr>
<td>X'14'</td>
<td>6 per row</td>
<td>151 modules</td>
</tr>
<tr>
<td>X'15'</td>
<td>8 per row</td>
<td>200 modules</td>
</tr>
<tr>
<td>X'16'</td>
<td>10 per row</td>
<td>249 modules</td>
</tr>
<tr>
<td>X'17'</td>
<td>12 per row</td>
<td>298 modules</td>
</tr>
<tr>
<td>X'18'</td>
<td>14 per row</td>
<td>347 modules</td>
</tr>
<tr>
<td>X'19'</td>
<td>16 per row</td>
<td>396 modules</td>
</tr>
<tr>
<td>X'1A'</td>
<td>18 per row</td>
<td>445 modules</td>
</tr>
<tr>
<td>X'1B'</td>
<td>20 per row</td>
<td>494 modules</td>
</tr>
</tbody>
</table>

Note: To determine the target width of the symbol in inches for a particular modifier value, multiply (number of modules from the table) times (module width). For example, if modifier X'1A' is specified and the module width is 10 mils, the target symbol width is 445 * 0.010 = 4.45 inches. If instead modifier X'14' is specified, the target symbol width is 151 * 0.010 = 1.51 inches. The height of the stacked symbol depends on how much data is encoded and how many rows were used, but in general a wide symbol will have fewer rows and therefore be shorter than a narrow symbol.

The BCOCA receiver will encode the input data to determine how many symbol characters are needed and will then attempt to create an Expanded Stacked symbol that contains the requested number of symbol characters per row. The receiver must work within the constraints defined by the GS1 DataBar symbology:
- There can be between two and eleven rows for an Expanded Stacked symbol; an Expanded symbol has one row.
- Each row, except for the bottom row, must have an even number of symbol characters.
• The bottom row must contain a minimum of two symbol characters (with padding added to the last symbol character if necessary).

The BCOWA receiver will attempt to create an Expanded Stacked symbol for which each row contains the requested number of symbol characters. Depending on the number of actual symbol characters generated, the bottom row might be shorter than the others or there might be only one row (an Expanded symbol). When there is insufficient input data to generate the minimum required number of symbol characters, the BCOWA receiver will continue to pad the binary data with the B'00100' padding string until sufficient symbol characters are built (some of these might consist only of pad bits). For example, there must be at least two symbol characters in the bottom row and the encodation methods require at least four symbol characters.

The height of each row is 34 times the module width and there is a 3 module high separator pattern between each row. The total symbol height is a multiple of the module width, which is 34*(number of rows)+3*(number of separator patterns).

The format of the input data for this bar code symbol is exactly the same as for a GS1 DataBar Expanded symbol. The bar code receiver will compact the data, pad the binary data with the B'00100' padding string until sufficient symbol characters are built, create guard patterns, create data-character patterns, calculate a checksum, create finder patterns, and generate a GS1 DataBar Expanded Stacked bar code symbol (or an Expanded symbol if the requested number of symbol characters is larger than the number of generated symbol characters).

Note: The GS1 DataBar symbology is controlled by the GS1 standards organization and is described in GS1 General Specifications.
**Royal Mail Mailmark (modifier values X'00' and X'01')**

The Royal Mail Mailmark symbology is defined and used by Royal Mail Group Ltd. for intelligent mail tracking and reporting. This bar code is a four-state symbol that has two variations, each with a fixed number of bars:

1. **Barcode C (66 bars)** – A variable content code that has a unique identifier (ID) and includes Postcode and Delivery Point information.
2. **Barcode L (78 bars)** – A variable content code that has a unique ID and includes Postcode and Delivery Point information.

The Royal Mail Mailmark symbology limits the symbol size; therefore BSD element height, height multiplier, and wide-to-narrow ratio fields are not applicable to this symbology and are ignored by BCOCAs. The module width field allows for two symbol sizes (small and optimal); the small symbol ranges from 2.8 inches wide to 3.3 inches wide (depending on the symbol variation) and the optimal symbol ranges from 3.1 inches wide to 3.7 inches wide. Refer to [Table 13 on page 41](#) for the specific dimensions.

The input data for the bar code is alphanumeric and consists of the fields shown in the *Royal Mail Mailmark Definition Document*. The data is checked for validity, and exception condition EC-1204 exists if there is invalid, insufficient, or too much data.

Human-Readable Interpretation (HRI) is not used with the Royal Mail Mailmark symbol. There must be a 2 mm quiet zone surrounding the symbol (all four sides).

The origin of the bar code symbol is defined to be the top-left corner of an imaginary rectangle of minimum size that bounds the bar and space pattern. If any part of the symbol falls outside the bar code presentation space, exception ID EC-1100 exists.

The Royal Mail Mailmark type uses the following modifier values:

- **X'00'** Barcode C
  - Present a 66 bar, variable-content-code symbol that has a unique identifier (ID) and includes Postcode and Delivery Point information.

- **X'01'** Barcode L
  - Present a 78 bar, variable-content-code symbol that has a unique ID and includes Postcode and Delivery Point information.

**Notes:**

1. The four-state Mailmark™ symbology replaces an earlier version called RED TAG. The RED TAG symbology has been retired by Royal Mail and has also been deprecated in the BCOCAs architecture.

2. There is also a 2D variation of the Mailmark symbology, called Complex Mail Data Marks (CMDM), which is produced as a Data Matrix symbol. Refer to the *Royal Mail Mailmark Definition Document* for a description of Complex Mail Data Marks.
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/CGPC 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.

### Table 20. Check Digit Calculation Methods

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Modifier</th>
<th>In HRI?</th>
<th>Check Digit Calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Code 39 (3-of-9 Code), AIM USS-39</td>
<td>X'02'</td>
<td>Yes</td>
<td>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.</td>
</tr>
</tbody>
</table>
| 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:  
   \[(10 - (\text{sum modulo 10})) \text{ modulo 10}\] |
|      |                    |          |        | 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:  
   \[(\text{sum modulo 11})\]  
   The check digit as 11 minus the remainder is described by the following equation:  
   \[(11 - (\text{sum modulo 11})) \text{ modulo 11}\] |
Check Digit Calculation Methods

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Modifier</th>
<th>In HRI?</th>
<th>Check Digit Calculation</th>
</tr>
</thead>
</table>
| X'03' | UPC/CGPC Version A | X'00' | Yes | 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:  
\[(\text{sum modulo 11})\]  
The check digit as 11 minus the remainder is described by the following equation:  
\[(11 - (\text{sum modulo 11})) \text{ modulo 11}\] |
| 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, ITF-14, AIM USS-I 2/5 | X'02' – X'04' | 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:  
\[(16 - (\text{sum modulo 16})) \text{ modulo 16}\] |
## Check Digit Calculation Methods

### Table 20  Check Digit Calculation Methods (cont’d.)

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Modifier</th>
<th>In HRI?</th>
<th>Check Digit Calculation</th>
</tr>
</thead>
</table>
| X'11'| Code 128, GS1-128, Intelligent Mail Container Barcode, UCC/EAN 128, AIM USS-128 | X'02' – X'05' | 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 (deprecated) and PLANET (deprecated) | X'00' – X'04' | NA | The 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 and Dutch KIX | X'00' | NA | The 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 “13 and 0”, 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) modulo 19. |
| X'1C'| Data Matrix, GS1 DataMatrix (2D bar code) | X'00' | NA | The Data Matrix symbology uses a Reed-Solomon Error Checking and Correcting (ECC) algorithm. |
| X'1D'| MaxiCode (2D bar code) | X'00' | NA | The MaxiCode symbology uses a Reed-Solomon Error Checking and Correcting (ECC) algorithm. |
| X'1E'| PDF417 (2D bar code) | 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 (2D bar code) | X'02' | NA | The QR Code symbology uses a Reed-Solomon Error Checking and Correcting (ECC) algorithm. |
### Table 20  Check Digit Calculation Methods (cont'd.)

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Modifier</th>
<th>In HRI?</th>
<th>Check Digit Calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'21'</td>
<td>Code 93</td>
<td>X'00'</td>
<td>No</td>
<td>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.</td>
</tr>
<tr>
<td>X'22'</td>
<td>Intelligent Mail Barcode</td>
<td>X'00' – X'03'</td>
<td>No</td>
<td>There is no check digit, but error detection and correction is added as part of the encoding process. Refer to United States Postal Service Specification USPS-B-3200, <em>Barcode, 4-State Customer</em>.</td>
</tr>
<tr>
<td>X'23'</td>
<td>Royal Mail RED TAG (deprecated)</td>
<td>X'00'</td>
<td>No</td>
<td>There is no check digit, but error detection and correction is added as part of the encoding process. Refer to <em>Royal Mail RED TAG Mailpiece Requirements</em>.</td>
</tr>
<tr>
<td>X'24'</td>
<td>GS1 DataBar</td>
<td>X'00' – X'04' and X'11' – X'1B'</td>
<td>No</td>
<td>There is no check digit, but an error detection checksum is calculated and is contained within the finder patterns. Refer to <em>GS1 General Specifications</em>.</td>
</tr>
<tr>
<td>X'25'</td>
<td>Royal Mail Mailmark</td>
<td>X'00'–X'01'</td>
<td>No</td>
<td>There is no check digit, but error detection and correction is added as part of the encoding process to ensure at least 25% error correction.</td>
</tr>
</tbody>
</table>
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:

Table 21. Bar Code Symbol Data (BSA) Data Structure

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>BITS</td>
<td>Bar code flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 0</td>
<td>HRI</td>
<td>B'0'</td>
<td>HRI is presented</td>
<td>B'0'</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B'1'</td>
<td>HRI not presented</td>
<td>B'1'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bits 1–2</td>
<td>Position</td>
<td>B'0'</td>
<td>Default</td>
<td>B'0'</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B'0'</td>
<td>HRI below</td>
<td>B'0'</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B'1'</td>
<td>HRI above</td>
<td>B'1'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 3</td>
<td>SSCAST</td>
<td>B'0'</td>
<td>Asterisk is not presented</td>
<td>B'0'</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B'1'</td>
<td>Asterisk is presented</td>
<td>B'1'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 4</td>
<td>B'0'</td>
<td>Retired item 21</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 5</td>
<td>SUPPRESS BAR CODE SYMBOL</td>
<td>B'0'</td>
<td>Bar code suppression:</td>
<td>B'0'</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B'1'</td>
<td>Present symbol</td>
<td>B'1'</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Suppress symbol</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 6</td>
<td>B'0'</td>
<td>Retired item 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B'1'</td>
<td>Desired method of adjusting for trailing blanks:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Suppess Suppess and adjust</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 7</td>
<td>X'0001'–X'7FFF'</td>
<td>X₀bc-coordinate of the symbol origin in the bar code presentation space</td>
<td>X'0001'–X'7FFF'</td>
<td>Refer to the note following the table.</td>
<td>X'0001'–X'7FFF'</td>
<td>Refer to the note following the table.</td>
</tr>
<tr>
<td>1–2</td>
<td>X offset</td>
<td>UBIN</td>
<td></td>
<td>X'0001'–X'7FFF'</td>
<td>X₀bc-coordinate of the symbol origin in the bar code presentation space</td>
<td>X'0001'–X'7FFF'</td>
</tr>
<tr>
<td>3–4</td>
<td>Y offset</td>
<td>UBIN</td>
<td></td>
<td>X'0001'–X'7FFF'</td>
<td>Y₀bc-coordinate of the symbol origin in the bar code presentation space</td>
<td>X'0001'–X'7FFF'</td>
</tr>
</tbody>
</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 | See field description |

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 | Any value defined for the bar code type selected by the BSD |

Note: The BCD1 and BCD2 range for these fields has been specified assuming a unit of measure of 1/1440 of an inch. Many receivers support the BCD1 or BCD2 subset plus additional function. If a receiver
supports additional units of measure, the BCOCA architecture requires the receiver to support a range equivalent to the subset 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 21.

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 are: Data Matrix, Japan Postal Bar Code, MaxiCode, PDF417, POSTNET (deprecated), QR Code, RM4SCC, Royal Mail RED TAG (deprecated), and Royal Mail Mailmark.

**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'.

- **B'00'** The presentation device default is used for positioning the HRI.
- **B'01'** The HRI is presented below the bar code symbol.
- **B'10'** The HRI is presented above the bar code symbol.
- **B'11'** Exception condition EC-1000 exists.

**Notes:**

1. HRI for GS1 DataBar and Intelligent Mail Container Barcode symbols must be positioned below the bar code symbol. The position flags (bits 1–2) are ignored for these symbols. HRI for Australia Post Bar Code should be positioned above the symbol.

2. 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.

3. If either the UPC or EAN Two-Digit and Five-Digit Supplemental bar code is selected in the BSD TYPE field (X'06', X'07', 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').
Bar Code Symbol Data (BSA)

**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**  
Retired item 21

**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 X offset and Y offset 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:

<table>
<thead>
<tr>
<th>Resulting Data Length</th>
<th>Bar Code Type</th>
<th>Bar Code Modifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>X'16' – Two-Digit Supplemental</td>
<td>X'00'</td>
</tr>
<tr>
<td>5</td>
<td>X'17' – Five-Digit Supplemental</td>
<td>X'00'</td>
</tr>
<tr>
<td>7</td>
<td>X'08' – EAN-8</td>
<td>X'00'</td>
</tr>
<tr>
<td>12</td>
<td>X'09' – EAN-13</td>
<td>X'00'</td>
</tr>
<tr>
<td>14</td>
<td>X'16' – Two-Digit Supplemental</td>
<td>X'01'</td>
</tr>
<tr>
<td>17</td>
<td>X'17' – Five-Digit Supplemental</td>
<td>X'01'</td>
</tr>
<tr>
<td>any other value</td>
<td>X'16' – Two-Digit Supplemental</td>
<td>X'01'</td>
</tr>
</tbody>
</table>

### 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:

<table>
<thead>
<tr>
<th>Resulting Data Length</th>
<th>Bar Code Type</th>
<th>Bar Code Modifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>X'06' – Two-Digit Supplemental</td>
<td>X'00'</td>
</tr>
<tr>
<td>5</td>
<td>X'07' – Five-Digit Supplemental</td>
<td>X'00'</td>
</tr>
<tr>
<td>10</td>
<td>X'05' – UPC version E</td>
<td>X'00'</td>
</tr>
<tr>
<td>11</td>
<td>X'03' – UPC version A</td>
<td>X'00'</td>
</tr>
<tr>
<td>12</td>
<td>X'06' – Two-Digit Supplemental</td>
<td>X'02'</td>
</tr>
<tr>
<td>13</td>
<td>X'06' – Two-Digit Supplemental</td>
<td>X'01'</td>
</tr>
<tr>
<td>15</td>
<td>X'07' – Five-Digit Supplemental</td>
<td>X'02'</td>
</tr>
<tr>
<td>16</td>
<td>X'07' – Five-Digit Supplemental</td>
<td>X'01'</td>
</tr>
<tr>
<td>any other value</td>
<td>X'06' – Two-Digit Supplemental</td>
<td>X'02'</td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>Resulting Data Length</th>
<th>Bar Code Type</th>
<th>Bar Code Modifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>X'18' – POSTNET</td>
<td>X'00'</td>
</tr>
<tr>
<td>9</td>
<td>X'18' – POSTNET</td>
<td>X'01'</td>
</tr>
<tr>
<td>11</td>
<td>X'18' – POSTNET</td>
<td>X'03'</td>
</tr>
</tbody>
</table>

If X'02' or X'04' was specified, that value is used; if any other modifier was specified, X'02' is used.

If X'18' – POSTNET
Bar Code Symbol Data (BSA)

If the user specified an Intelligent Mail Barcode type (X'22'):
Truncate the data and set the bar code type and modifier based on the resulting data length:

<table>
<thead>
<tr>
<th>Resulting Data Length</th>
<th>Bar Code Type</th>
<th>Bar Code Modifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>X'22' – Intelligent Mail Barcode</td>
<td>X'00'</td>
</tr>
<tr>
<td>25</td>
<td>X'22' – Intelligent Mail Barcode</td>
<td>X'01'</td>
</tr>
<tr>
<td>29</td>
<td>X'22' – Intelligent Mail Barcode</td>
<td>X'02'</td>
</tr>
<tr>
<td>31</td>
<td>X'22' – Intelligent Mail Barcode</td>
<td>X'03'</td>
</tr>
<tr>
<td>any other value</td>
<td>error</td>
<td></td>
</tr>
</tbody>
</table>

If the user specified any other bar code type:
Use the user-specified bar code type and modifier.

Bit 7 Retired item 3

Bytes 1–2 X offset
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 Xbc-coordinate of the top-left corner of an imaginary rectangle of minimum size that bounds the bar-space patterns (or two-dimensional module patterns) of the 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 Xbc-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 X offset value is invalid or unsupported.

Notes:

1. In most cases, the symbol origin is the top-left corner of the leftmost bar; however, this is not an appropriate origin for some bar code types, such as Dutch KIX, Intelligent Mail Barcode, MaxiCode, and Royal Mail Mailmark. The original BCoca symbol origin definition was the “top-left corner of the leftmost bar”; therefore, some older implementations might still use the original definition (this is not considered to be a deviation from the architecture for these older implementations).

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

3. For Royal Mail RED TAG (deprecated) symbols, use the top-left corner of the leftmost bar.

4. For GS1 DataBar symbols, the origin of the bar code symbol is the top-left corner of the leftmost space (since GS1 DataBar symbols begin with a space).
Bytes 3–4  
**Y offset**

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 an imaginary rectangle of minimum size that bounds the bar-space patterns (or two-dimensional module patterns) of the 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 $Y_{cr}$-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 Y offset value is invalid or unsupported.

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 94
- "MaxiCode Special-Function Parameters" on page 100
- "PDF417 Special-Function Parameters" on page 105
- "QR Code Special-Function Parameters" on page 111

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 can 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 139. 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 29 on page 120 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 28 on page 118 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 the documents listed in Table 5 on page xii.
## Data Matrix Special-Function Parameters

### Table 22. Data Matrix Special-Function Parameters

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>BITS</td>
<td>Control flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bit 0</td>
<td>EBCDIC</td>
<td>EBCDIC-to-ASCII translation:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'0'</td>
<td>Do not translate</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'1'</td>
<td>Convert data to ASCII</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bit 1</td>
<td>Escape</td>
<td>Escape-sequence handling:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sequence</td>
<td>Process escape sequences</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>handling</td>
<td>Ignore all escape sequences</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bits 2-7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'000000'</td>
<td>Reserved</td>
<td>B'000000'</td>
<td>B'000000'</td>
</tr>
<tr>
<td>6-7</td>
<td>UBIN</td>
<td>Desired row size</td>
<td>X'0000'</td>
<td>No size specified</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'0001'</td>
<td>Matrix row size as allowed by symbology; see field description</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'FFFF'</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-9</td>
<td>UBIN</td>
<td>Desired number of rows</td>
<td>X'0000'</td>
<td>No size specified</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'0001'</td>
<td>Number of rows as allowed by symbology; see field description</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'FFFF'</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>UBIN</td>
<td>Sequence indicator</td>
<td>X'00'–X'10'</td>
<td>Structured append sequence indicator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>UBIN</td>
<td>Total symbols</td>
<td>X'00'</td>
<td>Total number of structured-append symbols</td>
<td>X'00' or X'02'–X'10'</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X'02'</td>
<td></td>
<td>X'00' or X'02'–X'10'</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>UBIN</td>
<td>File ID 1st byte</td>
<td>X'01'–X'FE'</td>
<td>High-order byte of a 2-byte unique file identification for a set of structured-append symbols</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>UBIN</td>
<td>File ID 2nd byte</td>
<td>X'01'–X'FE'</td>
<td>Low-order byte of a 2-byte unique file identification for a set of structured-append symbols</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>BITS</td>
<td>Special-function flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bit 0</td>
<td>GS1 FNC1</td>
<td>Alternate data type identifier:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'0'</td>
<td>User-defined symbol</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'1'</td>
<td>Symbol conforms to GS1 standards</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bit 1</td>
<td>Industry FNC1</td>
<td>Alternate data type identifier:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'0'</td>
<td>User-defined symbol</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'1'</td>
<td>Symbol conforms to industry standards</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bit 2</td>
<td>Reader programming</td>
<td>Reader programming symbol:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'0'</td>
<td>Symbol encodes a data symbol</td>
<td>B'0'</td>
<td>B'1'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'1'</td>
<td>Symbol encodes a message used to program the reader system</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 22 Data Matrix Special-Function Parameters (cont’d.)

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>bits 3–4</td>
<td>Hdr/Ttl</td>
<td>Macro</td>
<td>B’00’</td>
<td>Header and trailer instructions to the bar code reader:</td>
<td>Not supported in BCD1</td>
<td>B’00’</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’01’</td>
<td>No header or trailer</td>
<td></td>
<td>B’01’</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’10’</td>
<td>Use the 05 Macro header/trailer</td>
<td></td>
<td>B’10’</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’11’</td>
<td>Use the 06 Macro header/trailer</td>
<td></td>
<td>B’11’</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>No header or trailer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bits 5–7</td>
<td>Encodation</td>
<td>scheme</td>
<td>B’000’</td>
<td>Encodation scheme used to produce the bar code symbol:</td>
<td>B’000’</td>
<td>B’000’</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’001’</td>
<td>Device default – usually Auto encoding</td>
<td>Other values not supported in BCD1</td>
<td>Other values not supported in BCD2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’010’</td>
<td>ASCII</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’011’</td>
<td>C40</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’100’</td>
<td>Text</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’101’</td>
<td>X12</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’110’</td>
<td>EDIFACT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B’111’</td>
<td>Base 256</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Auto encoding</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 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**
Data Matrix Special-Function Parameters

**Bytes 6–7** Desired row size

**Note:** A desired symbol size is 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.

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 23. Supported Sizes for a Data Matrix Symbol**

<table>
<thead>
<tr>
<th>Square Symbols</th>
<th>Data Region</th>
<th>Rectangular Symbols</th>
<th>Data Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Rows</td>
<td>Row Size</td>
<td>Size</td>
<td>Number</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>8x8</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>10x10</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>12x12</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>14x14</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>18</td>
<td>16x16</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td>18x18</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>22</td>
<td>20x20</td>
<td>1</td>
</tr>
<tr>
<td>24</td>
<td>24</td>
<td>22x22</td>
<td>1</td>
</tr>
<tr>
<td>26</td>
<td>26</td>
<td>24x24</td>
<td>1</td>
</tr>
<tr>
<td>32</td>
<td>32</td>
<td>14x14</td>
<td>4</td>
</tr>
<tr>
<td>36</td>
<td>36</td>
<td>16x16</td>
<td>4</td>
</tr>
<tr>
<td>40</td>
<td>40</td>
<td>18x18</td>
<td>4</td>
</tr>
<tr>
<td>44</td>
<td>44</td>
<td>20x20</td>
<td>4</td>
</tr>
<tr>
<td>48</td>
<td>48</td>
<td>22x22</td>
<td>4</td>
</tr>
<tr>
<td>52</td>
<td>52</td>
<td>24x24</td>
<td>4</td>
</tr>
<tr>
<td>64</td>
<td>64</td>
<td>14x14</td>
<td>16</td>
</tr>
<tr>
<td>72</td>
<td>72</td>
<td>16x16</td>
<td>16</td>
</tr>
<tr>
<td>80</td>
<td>80</td>
<td>18x18</td>
<td>16</td>
</tr>
<tr>
<td>88</td>
<td>88</td>
<td>20x20</td>
<td>16</td>
</tr>
<tr>
<td>96</td>
<td>96</td>
<td>22x22</td>
<td>16</td>
</tr>
<tr>
<td>104</td>
<td>104</td>
<td>24x24</td>
<td>16</td>
</tr>
<tr>
<td>120</td>
<td>120</td>
<td>18x18</td>
<td>36</td>
</tr>
</tbody>
</table>
Table 23  Supported Sizes for a Data Matrix Symbol (cont’d.)

<table>
<thead>
<tr>
<th>Square Symbols</th>
<th>Rectangular Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Symbol Size</strong></td>
<td><strong>Data Region</strong></td>
</tr>
<tr>
<td>Number of Rows</td>
<td>Row Size</td>
</tr>
<tr>
<td>132</td>
<td>132</td>
</tr>
<tr>
<td>144</td>
<td>144</td>
</tr>
</tbody>
</table>

**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-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, that 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.
Data Matrix Special-Function Parameters

**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**

GS1 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 GS1 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 GS1 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 GS1 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 GS1 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**

Encodation scheme used to produce bar code symbol

This field provides a means of selecting the encodation scheme used to produce the symbol. This is an optional special function that is not supported by all BCOCA receivers. Receivers that do not support this function, ignore these flags and use a device default method of choosing the encodation scheme. IPDS printers indicate support for this function with Sense Type and Model property pair X'1303'.

The selected encodation scheme is used for all of the data within the bar code object to produce a series of symbol data characters that are used to produce the bar code symbol. Usually the scheme is selected to produce the smallest number of symbol
data characters, but the best scheme might not be the one that produces the fewest bits per data character. Also, producing the fewest bits per data character might require switching between encodation schemes that can cause the symbol size to grow. The encodation schemes are described as follows:

**Device default (B'000')**

The BCOCA receiver uses a device-specific method of selecting and switching among encodation schemes. This is the scheme used by BCOCA receivers that ignore bits 5–7. Usually the device default is the same as auto encoding. If you are unsure of the encodation scheme to use, device default is a good choice.

**ASCII (B'001')**

This encodation scheme produces 4 bits per data character for double digit numerics, 8 bits per data character for ASCII values 0–127, and 16 bits per data character for Extended ASCII values 128–255.

**C40 (B'010')**

This encodation scheme is used when the input data is primarily upper-case alphanumeric and produces 5.33 bits per data character.

**Text (B'011')**

This encodation scheme is used when the input data is primarily lower-case alphanumeric and produces 5.33 bits per data character.

**X12 (B'100')**

This encodation scheme is used when the input data is defined with the ANSI X12 EDI data set and produces 5.33 bits per data character.

**EDIFACT (B'101')**

This encodation scheme is used when the input data is ASCII values 32–94 and produces 6 bits per data character.

**Base 256 (B'110')**

This encodation scheme is used when the data is binary (for example image or non-text data) and produces 8 bits per data character.

**Auto encoding (B'111')**

The BCOCA receiver starts with ASCII encodation and switches between encodation schemes as needed to produce the minimum symbol data characters. This algorithm is described in an Annex of *International Symbology Specification – Data Matrix*.

The C40, Text, X12, and EDIFACT encodation schemes do not support all 256 possible input characters. Exception condition EC-1201 exists if one of these encodation schemes is selected and an unsupported character is encountered in the bar code data.
MaxiCode Special-Function Parameters

Table 24. MaxiCode Special-Function Parameters

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>BITS</td>
<td>Control flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 0</td>
<td></td>
<td>EBCDIC</td>
<td>B'0'</td>
<td>EBCDIC-to-ASCII translation:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'1'</td>
<td>Do not translate</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Convert data to ASCII</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 1</td>
<td></td>
<td>Escape sequence</td>
<td>B'0'</td>
<td>Escape-sequence handling:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>handling</td>
<td>B'1'</td>
<td>Process escape sequences</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Ignore all escape sequences</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bits 2–7</td>
<td></td>
<td>Reserved</td>
<td>B'0000000'</td>
<td></td>
<td>B'0000000'</td>
<td>B'0000000'</td>
</tr>
<tr>
<td>6</td>
<td>CODE</td>
<td>Symbol mode</td>
<td>X'02'–X'06'</td>
<td>Mode 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mode 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mode 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mode 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mode 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>UBIN</td>
<td>Sequence indicator</td>
<td>X'00'–X'08'</td>
<td>Structured append sequence indicator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>UBIN</td>
<td>Total symbols</td>
<td>X'00' or X'02'–X'08'</td>
<td>Total number of structured-append symbols</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>BITS</td>
<td>Special-function flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 0</td>
<td></td>
<td>Zipper</td>
<td>B'0'</td>
<td>No zipper pattern</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B'1'</td>
<td>Vertical zipper pattern on right</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bits 1–7</td>
<td></td>
<td>Reserved</td>
<td>B'0000000'</td>
<td></td>
<td>B'0000000'</td>
<td>B'0000000'</td>
</tr>
</tbody>
</table>

Byte 5

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 encoding 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 barcode data consists of only characters from MaxiCode Code Set A, the number of codewords matches the number of barcode data characters. However, if the barcode data contains other characters, the number of codewords is greater than the number of barcode 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 barcode data consists of only characters from MaxiCode Code Set A, the number of codewords matches the number of barcode data characters. However, if the barcode data contains other characters, the number of codewords is greater than the number of barcode 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 barcode 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 barcode data consists of only characters from MaxiCode Code Set A, the number of codewords matches the number of barcode data characters. However, if the barcode data contains other characters, the number of codewords is greater than the number of barcode 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.
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).

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.

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

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 11 on page 104. 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 device resolution, then the pattern is repeated to yield an evenly spaced zipper pattern and contrast block.
MaxiCode Special-Function Parameters

Figure 11. Example of a MaxiCode Bar Code Symbol with Zipper and Contrast Block

Guideline Dimensions for the Zipper and Contrast Block

- **Contrast block anchor point**: This pattern repeats for a total of 9 bars, with each bar 2/203 inch by 28/203 inch. The space between each pair of bars is 1/203 inch. The contrast block anchor point is 38/203 inch directly above the zipper anchor point.

- **Zipper pattern anchor point**: This pattern repeats for approximately one inch (a total of 40 of these zipper teeth at 203 DPI). Each of the zipper teeth is made up of three 2/203 inch by 12/203 inch rectangles. The space between each pair of teeth is 1/203 inch. The zipper anchor point is 19/203 inch right of the rightmost column of hexagons that forms the MaxiCode symbol and is aligned with the top of the MaxiCode symbol.
## PDF417 Special-Function Parameters

### Table 25. PDF417 Special-Function Parameters

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>BITS</td>
<td>Control flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 0</td>
<td>EBCDIC</td>
<td>B'0' B'1'</td>
<td></td>
<td>EBCDIC-to-ASCII translation:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Do not translate</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Convert data to ASCII</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 1</td>
<td>Escape sequence handling</td>
<td>B'0' B'1'</td>
<td></td>
<td>Escape-sequence handling:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Process escape sequences</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Ignore all escape sequences</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bits 2–7</td>
<td></td>
<td>B'000000' Reserved</td>
<td></td>
<td></td>
<td>B'000000'</td>
<td>B'000000'</td>
</tr>
<tr>
<td>6</td>
<td>UBIN</td>
<td>Data symbols</td>
<td>X'01'–X'1E'</td>
<td>Number of data symbol characters per row</td>
<td>Not supported in BCD1</td>
<td>X'01'–X'1E'</td>
</tr>
<tr>
<td>7</td>
<td>UBIN</td>
<td>Rows</td>
<td>X'03'–X'5A' X'FF'</td>
<td>Desired number of rows</td>
<td>Not supported in BCD1</td>
<td>X'03'–X'5A' X'FF'</td>
</tr>
<tr>
<td>8</td>
<td>UBIN</td>
<td>Security</td>
<td>X'00'–X'08'</td>
<td>Security level</td>
<td>Not supported in BCD1</td>
<td>X'00'–X'08'</td>
</tr>
<tr>
<td>9–10</td>
<td>UBIN</td>
<td>Macro length</td>
<td>X'0000'–X'7FED'</td>
<td>Length of Macro PDF417 Control Block that follows</td>
<td>Not supported in BCD1</td>
<td>X'0000'–X'7FED'</td>
</tr>
<tr>
<td>11–n</td>
<td>UBIN</td>
<td>Macro data</td>
<td>Any value</td>
<td>Data for a Macro PDF417 Control Block</td>
<td>Not supported in BCD1</td>
<td>Any value</td>
</tr>
</tbody>
</table>

### 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 that include 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 12 on page 106 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:

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'10', 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'.

Figure 12. Subset of EBCDIC Code Page 500 That Can Be Translated to GLI 0

<table>
<thead>
<tr>
<th>Hex Digits</th>
<th>0-</th>
<th>1-</th>
<th>2-</th>
<th>3-</th>
<th>4-</th>
<th>5-</th>
<th>6-</th>
<th>7-</th>
<th>8-</th>
<th>9-</th>
<th>A-</th>
<th>B-</th>
<th>C-</th>
<th>D-</th>
<th>E-</th>
<th>F-</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NUL</td>
<td>DLE</td>
<td>(SP)</td>
<td>&amp;</td>
<td>_</td>
<td>°</td>
<td>µ</td>
<td>€</td>
<td>{</td>
<td>}</td>
<td>\</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>SOH</td>
<td>DC1</td>
<td>(RSP)</td>
<td>é</td>
<td>/</td>
<td>É</td>
<td>a</td>
<td>j</td>
<td>~</td>
<td>£</td>
<td>A</td>
<td>J</td>
<td>½</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>STX</td>
<td>DC2</td>
<td>SYN</td>
<td>á</td>
<td>è</td>
<td>b</td>
<td>k</td>
<td>s</td>
<td>¥</td>
<td>B</td>
<td>K</td>
<td>S</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ETX</td>
<td>DC3</td>
<td>ā</td>
<td>ē</td>
<td>Ň</td>
<td>Ä</td>
<td>c</td>
<td>l</td>
<td>t</td>
<td>C</td>
<td>L</td>
<td>T</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>HT</td>
<td>LF</td>
<td>á</td>
<td>è</td>
<td>d</td>
<td>m</td>
<td>u</td>
<td>D</td>
<td>M</td>
<td>U</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>BS</td>
<td>ETB</td>
<td>í</td>
<td>f</td>
<td>o</td>
<td>w</td>
<td>F</td>
<td>O</td>
<td>W</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>DEL</td>
<td>ESC</td>
<td>ñ</td>
<td>ï</td>
<td>Ň</td>
<td>ĝ</td>
<td>P</td>
<td>x</td>
<td>¼</td>
<td>G</td>
<td>P</td>
<td>X</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>CAN</td>
<td>SE330000</td>
<td>ç</td>
<td>i</td>
<td>Ć</td>
<td>h</td>
<td>q</td>
<td>y</td>
<td>½</td>
<td>H</td>
<td>Q</td>
<td>Y</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>EM</td>
<td>SE260000</td>
<td>Ņ</td>
<td>β</td>
<td>Ň</td>
<td>i</td>
<td>r</td>
<td>z</td>
<td>I</td>
<td>R</td>
<td>Z</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-A</td>
<td>[ ]</td>
<td>:</td>
<td>「</td>
<td>、</td>
<td>I</td>
<td>ŉ</td>
<td>ï</td>
<td>オ</td>
<td>オ</td>
<td>オ</td>
<td>オ</td>
<td>オ</td>
<td>オ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-B</td>
<td>VS</td>
<td>SE120000</td>
<td>、</td>
<td>$</td>
<td>#</td>
<td>»</td>
<td>ー</td>
<td>オ</td>
<td>オ</td>
<td>オ</td>
<td>オ</td>
<td>オ</td>
<td>オ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-C</td>
<td>FF</td>
<td>FS</td>
<td>DC4</td>
<td>&lt;</td>
<td>*</td>
<td>%</td>
<td>@</td>
<td>æ</td>
<td>◊</td>
<td>Ũ</td>
<td>Ũ</td>
<td>Ũ</td>
<td>Ũ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-D</td>
<td>CR</td>
<td>GS</td>
<td>ENQ</td>
<td>( )</td>
<td>-</td>
<td>)</td>
<td>)</td>
<td>)</td>
<td>)</td>
<td>)</td>
<td>)</td>
<td>)</td>
<td>)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-E</td>
<td>SO</td>
<td>RS</td>
<td>ACK</td>
<td>+</td>
<td>；</td>
<td>&gt;</td>
<td>=</td>
<td>Æ</td>
<td>．</td>
<td>．</td>
<td>．</td>
<td>．</td>
<td>．</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-F</td>
<td>SI</td>
<td>US</td>
<td>BEL</td>
<td>＃</td>
<td>？</td>
<td>&quot;</td>
<td>±</td>
<td>￥</td>
<td>￥</td>
<td>￥</td>
<td>￥</td>
<td>￥</td>
<td>￥</td>
<td>￥</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

PDF417 Special-Function Parameters
**PDF417 Special-Function Parameters**

**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**  
Reserved  
2–7

**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.
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:

<table>
<thead>
<tr>
<th>Security level</th>
<th>Maximum Limit of Erasures + 2*Misdecodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>4</td>
<td>30</td>
</tr>
<tr>
<td>5</td>
<td>62</td>
</tr>
<tr>
<td>6</td>
<td>126</td>
</tr>
<tr>
<td>7</td>
<td>254</td>
</tr>
<tr>
<td>8</td>
<td>510</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Number of Data Codewords</th>
<th>Recommended Security Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1–40</td>
<td>2</td>
</tr>
<tr>
<td>41–160</td>
<td>3</td>
</tr>
<tr>
<td>161–320</td>
<td>4</td>
</tr>
<tr>
<td>321–863</td>
<td>5</td>
</tr>
</tbody>
</table>

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

## Table 26. QR Code Special-Function Parameters

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>BITS</td>
<td>Control flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 0</td>
<td>EBCDIC</td>
<td>B'0'</td>
<td>B'0'</td>
<td>EBCDIC-to-ASCII translation:</td>
<td>Not supported in BCD1</td>
<td>B'0'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B'1'</td>
<td>B'1'</td>
<td>Do not translate</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Convert data to ASCII</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 1</td>
<td>Escape sequence handling</td>
<td>B'0'</td>
<td>B'0'</td>
<td>Escape-sequence handling:</td>
<td>Not supported in BCD1</td>
<td>B'0'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B'1'</td>
<td>B'1'</td>
<td>Process escape sequences</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Ignore all escape sequences</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bits 2–7</td>
<td>B'000000'</td>
<td>Reserved</td>
<td>B'000000'</td>
<td>No conversion specified</td>
<td>Not supported in BCD1</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td>CODE</td>
<td>Conversion</td>
<td>X'00'</td>
<td>SBCS EBCDIC code page used to encode data:</td>
<td>X'00'</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'01'</td>
<td></td>
<td>Code page 500 (International #5)</td>
<td>X'01'</td>
<td>X'01'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'02'</td>
<td></td>
<td>Code page 290 (Japanese Katakana Ext.)</td>
<td>X'02'</td>
<td>X'02'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'03'</td>
<td></td>
<td>Code page 1027 (Japanese Latin Extended)</td>
<td>X'03'</td>
<td>X'03'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'04'</td>
<td></td>
<td>AFP Line Data SOSI-data conversion:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'05'</td>
<td></td>
<td>CCSID 1390 to CCSID 943</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'06'</td>
<td></td>
<td>CCSID 1399 to CCSID 943</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'07'</td>
<td></td>
<td>CCSID 1390 to CCSID 932</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'08'</td>
<td></td>
<td>CCSID 1399 to CCSID 932</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'09'</td>
<td></td>
<td>CCSID 1390 to CCSID 942</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'0A'</td>
<td></td>
<td>CCSID 1399 to CCSID 942</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CODE</td>
<td>Version</td>
<td>X'00'</td>
<td>Version of symbol:</td>
<td>Not supported in BCD1</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'01' – X'28'</td>
<td></td>
<td>Smallest symbol</td>
<td>X'00'</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Version number (1 to 40)</td>
<td>X'01' – X'28'</td>
<td>X'01' – X'28'</td>
</tr>
<tr>
<td>8</td>
<td>CODE</td>
<td>Error correction level</td>
<td>X'00'</td>
<td>Level of error correction:</td>
<td>Not supported in BCD1</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'01'</td>
<td></td>
<td>Level L (7% recovery)</td>
<td>X'00'</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'02'</td>
<td></td>
<td>Level M (15% recovery)</td>
<td>X'01'</td>
<td>X'01'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'03'</td>
<td></td>
<td>Level Q (25% recovery)</td>
<td>X'02'</td>
<td>X'02'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Level H (30% recovery)</td>
<td>X'03'</td>
<td>X'03'</td>
</tr>
<tr>
<td>9</td>
<td>UBIN</td>
<td>Sequence indicator</td>
<td>X'00'</td>
<td>Structured append sequence indicator</td>
<td>Not supported in BCD1</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'10'</td>
<td></td>
<td></td>
<td>X'00' – X'10'</td>
<td>X'00' – X'10'</td>
</tr>
<tr>
<td>10</td>
<td>UBIN</td>
<td>Total symbols</td>
<td>X'00'</td>
<td>Total number of structured-append symbols</td>
<td>Not supported in BCD1</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'02' or X'10'</td>
<td></td>
<td></td>
<td>X'00' or X'02' – X'10'</td>
<td>X'00' or X'02' – X'10'</td>
</tr>
<tr>
<td>11</td>
<td>UBIN</td>
<td>Parity Data</td>
<td>X'00'</td>
<td>Structured append parity data</td>
<td>Not supported in BCD1</td>
<td>X'00'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X'FF'</td>
<td></td>
<td></td>
<td>X'00' – X'FF'</td>
<td>X'00' – X'FF'</td>
</tr>
<tr>
<td>12</td>
<td>BITS</td>
<td>Special-function flags</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 0</td>
<td>UCC/EAN FNC1</td>
<td>Alternate data type identifier:</td>
<td>B'0'</td>
<td>User-defined symbol</td>
<td>Not supported in BCD1</td>
<td>B'0'</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B'1'</td>
<td>B'1'</td>
<td>Symbol conforms to UCC/EAN standards</td>
<td></td>
<td>B'1'</td>
</tr>
</tbody>
</table>
# QR Code Special-Function Parameters

## Table 26  QR Code Special-Function Parameters (cont’d.)

<table>
<thead>
<tr>
<th>Bit 1</th>
<th>Industry FNC1</th>
<th>Alternate data type identifier:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>B'0' B'1'</td>
<td>User-defined symbol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbol conforms to industry</td>
</tr>
<tr>
<td></td>
<td></td>
<td>standards</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Not supported in BCD1</td>
</tr>
<tr>
<td></td>
<td>B'0' B'1'</td>
<td></td>
</tr>
</tbody>
</table>

| Bits 2–7 | Applicati-
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>on indicator</td>
</tr>
<tr>
<td></td>
<td>See field description</td>
</tr>
<tr>
<td></td>
<td>Application indicator for Industry FNC1</td>
</tr>
<tr>
<td></td>
<td>Not supported in BCD1</td>
</tr>
<tr>
<td></td>
<td>All values listed in the field description</td>
</tr>
</tbody>
</table>

### 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 a non-zero value is selected in byte 6, the EBCDIC input data will be converted into the default character encodation, as follows:
  - When the conversion parameter (byte 6) is X'01', X'02', or X'03', the BCOCA receiver will convert each byte of the bar code data from the EBCDIC single-byte code page specified in byte 6 into ASCII code page 897 before this data is used to build the bar code symbol. These conversion choices are supported by IPDS printers.
  - Conversion parameters X'04' – X'09' are defined for software products that build BCOCA bar codes from AFP Line Data (these values are not supported by IPDS printers). The AFP Line Data software will convert the input line data from EBCDIC SOSI data into mixed-byte ASCII as specified by the conversion parameter.
  - When the conversion parameter (byte 6) is X'00', no translation is done.

#### 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.

#### Bits 2–7  Reserved
Byte 6  Conversion

When the EBCDIC-to-ASCII translation flag is B'1', this parameter specifies the method used to convert EBCDIC input data into the default character encodation. When the EBCDIC-to-ASCII translation flag is B'0', this parameter is not used and should be set to X'00'.

For the first three values (used when the input data is encoded with a single-byte EBCDIC code page), this parameter identifies the EBCDIC code page that encodes single-byte EBCDIC bar code data. The following EBCDIC code pages are supported:

X'01'  Code page 500 (International #5)

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 13 on page 114.

X'02'  Code page 290 (Japanese Katakana Extended)

X'03'  Code page 1027 (Japanese Latin Extended)

For the remaining values (used when the input data is SOSI), this parameter identifies the desired conversion from EBCDIC SOSI input data to a specific mixed-byte ASCII encoding.

Note: The values X'04' through X'09' are defined for the Additional Bar Code Parameters (X'7B') triplet used with AFP Line Data; these values are not valid within a BCOCA object built for a non-line-data environment, such as MO:DCA and IPDS. Refer to the Advanced Function Presentation: Programming Guide and Line Data Reference for a description of the Additional Bar Code Parameters (X'7B') triplet.

The following choices are supported:

X'04'  CCSID 1390 to CCSID 943

Convert from:  CCSID 1390 – Extended Japanese Katakana-Kanji Host Mixed for JIS X0213 including 6205 UDC, Extended SBCS (includes SBCS & DBCS euro)

Convert to:  CCSID 943 – Japanese PC Data Mixed for Open environment (Multi-vendor code): 6878 JIS X 0208-1990 chars, 386 IBM selected DBCS chars, 1880 UDC (X'F040' to X'F9FC')

X'05'  CCSID 1399 to CCSID 943

Convert from:  CCSID 1399 – Extended Japanese Latin-Kanji Host Mixed for JIS X0213 including 6205 UDC, Extended SBCS (includes SBCS & DBCS euro)

Convert to:  CCSID 943 – Japanese PC Data Mixed for Open environment (Multi-vendor code): 6878 JIS X 0208-1990 chars, 386 IBM selected DBCS chars, 1880 UDC (X'F040' to X'F9FC')

X'06'  CCSID 1390 to CCSID 932

Convert from:  CCSID 1390 – Extended Japanese Katakana-Kanji Host Mixed for JIS X0213 including 6205 UDC, Extended SBCS (includes SBCS & DBCS euro)

Convert to:  CCSID 932 – Japanese PC Data Mixed including 1880 UDC

X'07'  CCSID 1399 to CCSID 932

Convert from:  CCSID 1399 – Extended Japanese Latin-Kanji Host Mixed for JIS X0213 including 6205 UDC, Extended SBCS (includes SBCS & DBCS euro)

Convert to:  CCSID 932 – Japanese PC Data Mixed including 1880 UDC
### X'08'  CCSID 1390 to CCSID 942
**Convert from:**  CCSID 1390 – Extended Japanese Katakana-Kanji Host Mixed for JIS X0213 including 6205 UDC, Extended SBCS (includes SBCS & DBCS euro)
**Convert to:**  CCSID 942 – Japanese PC Data Mixed including 1880 UDC, Extended SBCS

EBCDIC characters that are not defined within ECI 000020 are mapped to the substitute character, X'7F' or X'FCFC'; exception condition EC-2100 exists when an undefined character is encountered.

Exception condition EC-0F0E exists if an invalid or unsupported conversion value is specified.

---

### Figure 13. Subset of EBCDIC Code Page 500 That Can Be Translated To ECI 000020

<table>
<thead>
<tr>
<th>Hex Digit</th>
<th>0-</th>
<th>1-</th>
<th>2-</th>
<th>3-</th>
<th>4-</th>
<th>5-</th>
<th>6-</th>
<th>7-</th>
<th>8-</th>
<th>9-</th>
<th>A-</th>
<th>B-</th>
<th>C-</th>
<th>D-</th>
<th>E-</th>
<th>F-</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NUL</td>
<td>DLE</td>
<td>(SP)</td>
<td>&amp;</td>
<td>S-</td>
<td>0</td>
<td>S-</td>
<td>1</td>
<td>S-</td>
<td>2</td>
<td>S-</td>
<td>3</td>
<td>S-</td>
<td>4</td>
<td>S-</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>SOH</td>
<td>DC1</td>
<td>/</td>
<td>\</td>
<td></td>
<td></td>
<td>A</td>
<td>J</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>STX</td>
<td>DC2</td>
<td>b</td>
<td>k</td>
<td>s</td>
<td>¥</td>
<td>B</td>
<td>K</td>
<td>S</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ETX</td>
<td>DC3</td>
<td>c</td>
<td>l</td>
<td>t</td>
<td>C</td>
<td>L</td>
<td>T</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>d</td>
<td>m</td>
<td>u</td>
<td></td>
<td></td>
<td>D</td>
<td>M</td>
<td>U</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>e</td>
<td>n</td>
<td>v</td>
<td></td>
<td></td>
<td>E</td>
<td>N</td>
<td>V</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>f</td>
<td>o</td>
<td>w</td>
<td></td>
<td></td>
<td>F</td>
<td>O</td>
<td>W</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>g</td>
<td>P</td>
<td>x</td>
<td>G</td>
<td>P</td>
<td>X</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>h</td>
<td>q</td>
<td>y</td>
<td>H</td>
<td>Q</td>
<td>Y</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>i</td>
<td>r</td>
<td>z</td>
<td>I</td>
<td>R</td>
<td>Z</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Version of symbol

Note: 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.

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 27. Supported Versions for a QR Code Symbol

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

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.
QR Code Special-Function Parameters

**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'.

116  BCOCA Reference
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 Reserved
2–7

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>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>CPGID</th>
<th>FGID</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Code 39 (3-of-9 Code), AIM USS-39</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'02'</td>
<td>MSI (modified Plessey code)</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'03'</td>
<td>UPC/CGPC – Version A</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'05'</td>
<td>UPC/CGPC – Version E</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'06'</td>
<td>UPC – Two-Digit Supplemental (Periodicals)</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'07'</td>
<td>UPC – Five-Digit Supplemental (Paperbacks)</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'08'</td>
<td>EAN-8 (includes JAN-short)</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'09'</td>
<td>EAN-13 (includes JAN-standard)</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'0A'</td>
<td>Industrial 2-of-5</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'0B'</td>
<td>Matrix 2-of-5</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'0C'</td>
<td>Interleaved 2-of-5, ITF-14, AIM USS-I 2/5</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'0D'</td>
<td>Codabar, 2-of-7, AIM USS-Codabar</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'16'</td>
<td>EAN Two-Digit Supplemental</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'17'</td>
<td>EAN Five-Digit Supplemental</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'18'</td>
<td>POSTNET <em>(deprecated)</em> and PLANET <em>(deprecated)</em></td>
<td>500</td>
<td>None</td>
</tr>
<tr>
<td>X'1A'</td>
<td>RM4SCC and Dutch KIX</td>
<td>500</td>
<td>None</td>
</tr>
<tr>
<td>X'1B'</td>
<td>Japan Postal Bar Code</td>
<td>500</td>
<td>None</td>
</tr>
<tr>
<td>X'1C'</td>
<td>Data Matrix, GS1 DataMatrix (2D bar code)</td>
<td>Default CPGID=819; code page is selectable within the symbol using ECI protocol</td>
<td>None</td>
</tr>
<tr>
<td>X'1D'</td>
<td>MaxiCode (2D bar code)</td>
<td>Default CPGID=819; code page is selectable within the symbol using ECI protocol</td>
<td>None</td>
</tr>
<tr>
<td>X'1E'</td>
<td>PDF417 (2D bar code)</td>
<td>Default CPGID=437; code page is selectable within the symbol using GLI protocol</td>
<td>None</td>
</tr>
<tr>
<td>X'1F'</td>
<td>Australia Post Bar Code</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'20'</td>
<td>QR Code (2D bar code)</td>
<td>Default CPGID=897; code page is selectable within the symbol using ECI protocol</td>
<td>None</td>
</tr>
<tr>
<td>X'21'</td>
<td>Code 93</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'22'</td>
<td>Intelligent Mail Barcode</td>
<td>500</td>
<td>Device specific</td>
</tr>
</tbody>
</table>
**Table 28  Valid Code Pages and Type Styles (cont'd.)**

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>CPGID</th>
<th>FGID</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'23'</td>
<td>Royal Mail RED TAG (deprecated)</td>
<td>500</td>
<td>None</td>
</tr>
<tr>
<td>X'24'</td>
<td>GS1 DataBar</td>
<td>1303</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'25'</td>
<td>Royal Mail Mailmark</td>
<td>500</td>
<td>None</td>
</tr>
</tbody>
</table>

**Note:** Some symbologies allow a variety of FGIDs, but individual printers restrict the choice; when “Device specific” is specified in the FGID column, refer to printer documentation for information about supported FGIDs.
Valid Characters and Data Lengths

Table 29 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/CGPC 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 139.

Table 29. Valid Characters and Data Lengths

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Valid Characters</th>
<th>Valid Data Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Code 39 (3-of-9 Code), AIM USS-39</td>
<td>0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ -/$+%(space)</td>
<td>Symbology: unlimited BCOCA range: 0 to 50 characters (see note 2 on page 124)</td>
</tr>
<tr>
<td>X'02'</td>
<td>MSI (modified Plessey code)</td>
<td>0123456789</td>
<td>3 to 15 characters for Modifier X'01' 2 to 14 characters for Modifier X'02' 1 to 13 characters for all other modifiers</td>
</tr>
<tr>
<td>X'03'</td>
<td>UPC/CGPC - Version A</td>
<td>0123456789 (see note 1 on page 124)</td>
<td>11 characters</td>
</tr>
<tr>
<td>X'05'</td>
<td>UPC/CGPC - Version E</td>
<td>0123456789 (see note 1 on page 124)</td>
<td>10 characters</td>
</tr>
<tr>
<td>X'06'</td>
<td>UPC - Two-Digit Supplemental (Periodicals)</td>
<td>0123456789</td>
<td>2 characters for Modifier X'00' 13 characters for Modifier X'01' 12 characters for Modifier X'02'</td>
</tr>
<tr>
<td>X'07'</td>
<td>UPC - Five-Digit Supplemental (Paperbacks)</td>
<td>0123456789</td>
<td>5 characters for Modifier X'00' 16 characters for Modifier X'01' 15 characters for Modifier X'02'</td>
</tr>
<tr>
<td>X'08'</td>
<td>EAN-8 (includes JAN-short)</td>
<td>0123456789 (see note 1 on page 124)</td>
<td>7 characters</td>
</tr>
<tr>
<td>X'09'</td>
<td>EAN-13 (includes JAN-standard)</td>
<td>0123456789 (see note 1 on page 124)</td>
<td>12 characters</td>
</tr>
<tr>
<td>X'0A'</td>
<td>Industrial 2-of-5</td>
<td>0123456789</td>
<td>Symbology: unlimited BCOCA range: 0 to 50 characters (see note 2 on page 124)</td>
</tr>
<tr>
<td>X'0B'</td>
<td>Matrix 2-of-5</td>
<td>0123456789</td>
<td>Symbology: unlimited BCOCA range: 0 to 50 characters (see note 2 on page 124)</td>
</tr>
<tr>
<td>X'0C'</td>
<td>Interleaved 2-of-5, ITF-14, AIM USS-1 2/5</td>
<td>0123456789</td>
<td>Interleaved 2-of-5 symbology: unlimited ITF-14 symbology: 13 digits BCOCA range: 0 to 50 characters (see note 2 on page 124)</td>
</tr>
<tr>
<td>Type</td>
<td>Bar Code Symbology</td>
<td>Valid Characters</td>
<td>Valid Data Length</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| X’0D’ | Codabar, 2-of-7, AIM USS-Codabar    | 0123456789 -$.:/+.ABCD 16 characters plus 4 start/stop characters (ABCD) (see note 3 on page 124) | Symbology: unlimited  
BCOCA range: 0 to 50 characters (see note 124)                                    |
| X’11’ | Code 128, AIM USS-128 (modifier X’02’) | 0123456789 -$.:/+.ABCD 16 characters plus 4 start/stop characters (ABCD) (see note 3 on page 124) | Symbology: unlimited  
BCOCA range: 0 to 50 characters (see note 124)                                    |
| UCC/EAN 128 (modifier X’03’) | 0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ  
abcdefghijklmnopqrstuvwxyz  
FNC1 (X’8F’)) & All characters defined in the Code 128 code page (see page 129) | Maximum of 48 characters (see note 4 on page 124) |
| UCC/EAN 128, GS1-128 (modifier X’04’) | 0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ  
abcdefghijklmnopqrstuvwxyz  
FNC1 (X’8F’)) & All characters defined in the Code 128 code page (see page 129) | Maximum of 48 characters (see note 4 on page 124) |
| Intelligent Mail Container Barcode (modifier X’05’) | 0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ  
abcdefghijklmnopqrstuvwxyz  
FNC1 (X’8F’)) & Some fields restrict the range of characters; refer to the modifier X’05’ description in Table 15 on page 60. | 22 characters |
| X’16’ | EAN Two-Digit Supplemental         | 0123456789 2 characters for Modifier X’00’  
14 characters for Modifier X’01’ | 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’ | 5 characters for Modifier X’00’  
17 characters for Modifier X’01’ |
| X’18’ | POSTNET (deprecated) and PLANET (deprecated) | 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’ | 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 2 on page 124) |
| X’1A’ | Royal Mail (RM4SCC, modifier X’00’) | 0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ  
Symbology: unlimited  
BCOCA range: 0 to 50 characters (see note 2 on page 124) |
## Valid Characters and Data Lengths

Table 29  Valid Characters and Data Lengths (cont’d.)

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Valid Characters</th>
<th>Valid Data Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Royal Mail (Dutch KIX variation, modifier X’01’)</td>
<td>0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz</td>
<td>Symbology: maximum of 18 characters  BCOCA range: 0 to 50 characters (see note 2 on page 124)</td>
<td></td>
</tr>
<tr>
<td>X’1B’ Japan Postal Bar Code (Modifier X’00’)</td>
<td>0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ -</td>
<td>Symbology: 7 or more BCOCA range: 7 to 50 characters (see note 2 on page 124)</td>
<td></td>
</tr>
<tr>
<td>Japan Postal Bar Code (Modifier X’01’)</td>
<td>0123456789 CC1,CC2,CC3,CC4, CC5,CC6,CC7,CC8 - start, stop</td>
<td>No length checking done; refer to the modifier X’01’ description</td>
<td></td>
</tr>
<tr>
<td>X’1C’ Data Matrix, GS1 DataMatrix</td>
<td>Any one-byte character or binary data</td>
<td>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 2 on page 124)</td>
<td></td>
</tr>
<tr>
<td>X’1D’ MaxiCode</td>
<td>Any one-byte character allowed by the symbol mode; refer to the description of symbol modes on page 101</td>
<td>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</td>
<td></td>
</tr>
<tr>
<td>X’1E’ PDF417</td>
<td>Any one-byte character or binary data</td>
<td>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</td>
<td></td>
</tr>
<tr>
<td>X’1F’ Australia Post Bar Code – refer to the modifier (byte 13) description in “Australia Post Bar Code (modifier values X’01’ through X’08’)” on page 72 for information about valid characters in specific parts of the symbol</td>
<td>0123456789</td>
<td>8 digits</td>
<td></td>
</tr>
<tr>
<td>Modifier X’01’ – Standard Customer Barcode</td>
<td>0123456789</td>
<td>8 digits</td>
<td></td>
</tr>
<tr>
<td>Modifier X’02’ – Customer Barcode 2 using Table N</td>
<td>0123456789</td>
<td>8–16 digits</td>
<td></td>
</tr>
<tr>
<td>Modifier X’03’ – Customer Barcode 2 using Table C</td>
<td>0123456789</td>
<td>8–13 characters</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Bar Code Symbolology</td>
<td>Valid Characters</td>
<td>Valid Data Length</td>
</tr>
<tr>
<td>------</td>
<td>----------------------</td>
<td>-----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Modifier X'04' – Customer Barcode 2 using proprietary encoding</td>
<td>0123456789 for sorting code 0–3 for customer information</td>
<td>8–24 digits</td>
<td></td>
</tr>
<tr>
<td>Modifier X'05' – Customer Barcode 3 using Table N</td>
<td>0123456789</td>
<td>8–23 digits</td>
<td></td>
</tr>
<tr>
<td>Modifier X'06' – Customer Barcode 3 using Table C</td>
<td>0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz (space) #</td>
<td>8–18 characters</td>
<td></td>
</tr>
<tr>
<td>Modifier X'07' – Customer Barcode 3 using proprietary encoding</td>
<td>0123456789 for sorting code 0–3 for customer information</td>
<td>8–39 digits</td>
<td></td>
</tr>
<tr>
<td>Modifier X'08' – Reply Paid Barcode</td>
<td>0123456789</td>
<td>8 digits</td>
<td></td>
</tr>
<tr>
<td>X'20'</td>
<td>QR Code</td>
<td>Any one-byte character or binary data</td>
<td>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</td>
</tr>
<tr>
<td>X'21'</td>
<td>Code 93</td>
<td>0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ -$/+% (space) a (representing Shift 1) b (representing Shift 2) c (representing Shift 3) d (representing Shift 4) A total of 47 valid input characters</td>
<td>Symbology: unlimited BCOCA range: 0 to 50 characters (see note 2 on page 124)</td>
</tr>
<tr>
<td>X'22'</td>
<td>Intelligent Mail Barcode</td>
<td>0123456789</td>
<td>20 digits for Modifier X'00' 25 digits for Modifier X'01' 29 digits for Modifier X'02' 31 digits for Modifier X'03'</td>
</tr>
<tr>
<td>X'23'</td>
<td>Royal Mail RED TAG (deprecated)</td>
<td>0123456789</td>
<td>21 digits</td>
</tr>
<tr>
<td>X'24'</td>
<td>GS1 DataBar</td>
<td>0123456789</td>
<td>14 digits</td>
</tr>
<tr>
<td>Omnidirectional (Modifier X'00')</td>
<td>0123456789</td>
<td>14 digits</td>
<td></td>
</tr>
<tr>
<td>Truncated (Modifier X'01')</td>
<td>0123456789</td>
<td>14 digits</td>
<td></td>
</tr>
<tr>
<td>Stacked (Modifier X'02')</td>
<td>0123456789</td>
<td>14 digits</td>
<td></td>
</tr>
</tbody>
</table>
### Table 29  Valid Characters and Data Lengths (cont’d.)

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>Valid Characters</th>
<th>Valid Data Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stacked Omnidirectional (Modifier X'03')</td>
<td>0123456789</td>
<td>14 digits</td>
<td></td>
</tr>
<tr>
<td>Limited (Modifier X'04')</td>
<td>0123456789</td>
<td>14 digits</td>
<td></td>
</tr>
<tr>
<td>The first digit must be 0 or 1.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expanded (Modifier X'11')</td>
<td>0123456789</td>
<td>up to 74 digits or up to 41 alphabetic characters</td>
<td></td>
</tr>
<tr>
<td>ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz !&quot;%&amp;'()*+,-/:;&lt;&gt;?_ FNC1 (X'8F')</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expanded Stacked (Modifiers X'12' through X'1B')</td>
<td>0123456789</td>
<td>up to 74 digits or up to 41 alphabetic characters</td>
<td></td>
</tr>
<tr>
<td>ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz !&quot;%&amp;'()*+,-/:;&lt;&gt;?_ FNC1 (X'8F')</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X'25' Royal Mail Mailmark</td>
<td>0123456789</td>
<td>22 characters</td>
<td></td>
</tr>
<tr>
<td>Barcode C (Modifier X'00')</td>
<td>0123456789</td>
<td>22 characters</td>
<td></td>
</tr>
<tr>
<td>ABCDEFGHIJKLMNOPQRSTUVWXYZ (space)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Barcode L (Modifier X'01')</td>
<td>0123456789</td>
<td>26 characters</td>
<td></td>
</tr>
<tr>
<td>ABCDEFGHIJKLMNOPQRSTUVWXYZ (space)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Notes:

1. 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 description of GS1 company prefixes in the *GS1 General Specifications* for more details.

2. All BCOCAs must support at least the BCOC range. Some receivers support a larger data length.

3. 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 BCOC Codabar symbol, start and stop characters must be represented using A, B, C, or D.

4. A full description of the GS1-128 symbology is available in *GS1 General Specifications*. This document extends some of the Application Identifiers (AI) defined for UCC/EAN 128 to also allow 20 punctuation characters – !"%&'()*+,-/:;<>?_ – for GS1-128 symbols. The document also lists the following symbol size
characteristics for GS1-128 bar codes (but many BCOCAs receivers that support modifiers X'03' and X'04' do not check for or enforce these limits):

1. The maximum number of data characters in a single symbol is 48.
2. The maximum physical length of a Code 128 symbol is 165 mm (6.5 inches) including quiet zones.
**Characters and Code Points**

The following table (Table 30) is informational and is not a formal part of the BCOCA architecture. The table is intended as a convenient listing of some EBCDIC and ASCII code points and is not intended to be complete or to show all possible EBCDIC or ASCII encodings for any particular code point. The specific code pages are listed, using CPGIDs, in Table 28 on page 118. For a formal definition of these codes pages and CPGIDs, refer to the Character Data Representation Architecture listed in Table 5 on page xii. Note that this table does not necessarily cover all of the code points used for 2D bar codes and does not contain all of the characters available with CPGID = 1303.

Table 30. Characters and Code Points Commonly used in the BCOCA Symbologies (Not a Complete Listing)

<table>
<thead>
<tr>
<th>Character</th>
<th>EBCDIC Code Point</th>
<th>ASCII Code Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X'F0'</td>
<td>X'30'</td>
</tr>
<tr>
<td>1</td>
<td>X'F1'</td>
<td>X'31'</td>
</tr>
<tr>
<td>2</td>
<td>X'F2'</td>
<td>X'32'</td>
</tr>
<tr>
<td>3</td>
<td>X'F3'</td>
<td>X'33'</td>
</tr>
<tr>
<td>4</td>
<td>X'F4'</td>
<td>X'34'</td>
</tr>
<tr>
<td>5</td>
<td>X'F5'</td>
<td>X'35'</td>
</tr>
<tr>
<td>6</td>
<td>X'F6'</td>
<td>X'36'</td>
</tr>
<tr>
<td>7</td>
<td>X'F7'</td>
<td>X'37'</td>
</tr>
<tr>
<td>8</td>
<td>X'F8'</td>
<td>X'38'</td>
</tr>
<tr>
<td>9</td>
<td>X'F9'</td>
<td>X'39'</td>
</tr>
<tr>
<td>A</td>
<td>X'C1'</td>
<td>X'41'</td>
</tr>
<tr>
<td>B</td>
<td>X'C2'</td>
<td>X'42'</td>
</tr>
<tr>
<td>C</td>
<td>X'C3'</td>
<td>X'43'</td>
</tr>
<tr>
<td>D</td>
<td>X'C4'</td>
<td>X'44'</td>
</tr>
<tr>
<td>E</td>
<td>X'C5'</td>
<td>X'45'</td>
</tr>
<tr>
<td>F</td>
<td>X'C6'</td>
<td>X'46'</td>
</tr>
<tr>
<td>G</td>
<td>X'C7'</td>
<td>X'47'</td>
</tr>
<tr>
<td>H</td>
<td>X'C8'</td>
<td>X'48'</td>
</tr>
<tr>
<td>I</td>
<td>X'C9'</td>
<td>X'49'</td>
</tr>
<tr>
<td>J</td>
<td>X'D1'</td>
<td>X'4A'</td>
</tr>
<tr>
<td>K</td>
<td>X'D2'</td>
<td>X'4B'</td>
</tr>
<tr>
<td>L</td>
<td>X'D3'</td>
<td>X'4C'</td>
</tr>
<tr>
<td>M</td>
<td>X'D4'</td>
<td>X'4D'</td>
</tr>
<tr>
<td>N</td>
<td>X'D5'</td>
<td>X'4E'</td>
</tr>
<tr>
<td>O</td>
<td>X'D6'</td>
<td>X'4F'</td>
</tr>
<tr>
<td>P</td>
<td>X'D7'</td>
<td>X'50'</td>
</tr>
<tr>
<td>Q</td>
<td>X'D8'</td>
<td>X'51'</td>
</tr>
<tr>
<td>R</td>
<td>X'D9'</td>
<td>X'52'</td>
</tr>
<tr>
<td>S</td>
<td>X'E2'</td>
<td>X'53'</td>
</tr>
<tr>
<td>T</td>
<td>X'E3'</td>
<td>X'54'</td>
</tr>
<tr>
<td>U</td>
<td>X'E4'</td>
<td>X'55'</td>
</tr>
<tr>
<td>Character</td>
<td>EBCDIC Code Point</td>
<td>ASCII Code Point</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>V</td>
<td>X'E5'</td>
<td>X'56'</td>
</tr>
<tr>
<td>W</td>
<td>X'E6'</td>
<td>X'57'</td>
</tr>
<tr>
<td>X</td>
<td>X'E7'</td>
<td>X'58'</td>
</tr>
<tr>
<td>Y</td>
<td>X'E8'</td>
<td>X'59'</td>
</tr>
<tr>
<td>Z</td>
<td>X'E9'</td>
<td>X'5A'</td>
</tr>
<tr>
<td>a</td>
<td>X'81'</td>
<td>X'61'</td>
</tr>
<tr>
<td>b</td>
<td>X'82'</td>
<td>X'62'</td>
</tr>
<tr>
<td>c</td>
<td>X'83'</td>
<td>X'63'</td>
</tr>
<tr>
<td>d</td>
<td>X'84'</td>
<td>X'64'</td>
</tr>
<tr>
<td>e</td>
<td>X'85'</td>
<td>X'65'</td>
</tr>
<tr>
<td>f</td>
<td>X'86'</td>
<td>X'66'</td>
</tr>
<tr>
<td>g</td>
<td>X'87'</td>
<td>X'67'</td>
</tr>
<tr>
<td>h</td>
<td>X'88'</td>
<td>X'68'</td>
</tr>
<tr>
<td>i</td>
<td>X'89'</td>
<td>X'69'</td>
</tr>
<tr>
<td>j</td>
<td>X'91'</td>
<td>X'6A'</td>
</tr>
<tr>
<td>k</td>
<td>X'92'</td>
<td>X'6B'</td>
</tr>
<tr>
<td>l</td>
<td>X'93'</td>
<td>X'6C'</td>
</tr>
<tr>
<td>m</td>
<td>X'94'</td>
<td>X'6D'</td>
</tr>
<tr>
<td>n</td>
<td>X'95'</td>
<td>X'6E'</td>
</tr>
<tr>
<td>o</td>
<td>X'96'</td>
<td>X'6F'</td>
</tr>
<tr>
<td>p</td>
<td>X'97'</td>
<td>X'70'</td>
</tr>
<tr>
<td>q</td>
<td>X'98'</td>
<td>X'71'</td>
</tr>
<tr>
<td>r</td>
<td>X'99'</td>
<td>X'72'</td>
</tr>
<tr>
<td>s</td>
<td>X'A2'</td>
<td>X'73'</td>
</tr>
<tr>
<td>t</td>
<td>X'A3'</td>
<td>X'74'</td>
</tr>
<tr>
<td>u</td>
<td>X'A4'</td>
<td>X'75'</td>
</tr>
<tr>
<td>v</td>
<td>X'A5'</td>
<td>X'76'</td>
</tr>
<tr>
<td>w</td>
<td>X'A6'</td>
<td>X'77'</td>
</tr>
<tr>
<td>x</td>
<td>X'A7'</td>
<td>X'78'</td>
</tr>
<tr>
<td>y</td>
<td>X'A8'</td>
<td>X'79'</td>
</tr>
<tr>
<td>z</td>
<td>X'A9'</td>
<td>X'7A'</td>
</tr>
<tr>
<td>-</td>
<td>X'60'</td>
<td>X'2D'</td>
</tr>
<tr>
<td>#</td>
<td>X'7B'</td>
<td>X'23'</td>
</tr>
<tr>
<td>.</td>
<td>X'4B'</td>
<td>X'2E'</td>
</tr>
<tr>
<td>$</td>
<td>X'5B'</td>
<td>X'24'</td>
</tr>
<tr>
<td>/</td>
<td>X'61'</td>
<td>X'2F'</td>
</tr>
<tr>
<td>+</td>
<td>X'4E'</td>
<td>X'2B'</td>
</tr>
<tr>
<td>%</td>
<td>X'6C'</td>
<td>X'25'</td>
</tr>
</tbody>
</table>
### Characters and Code Points

Table 30  **Characters and Code Points Commonly used in the BCOCA Symbologies (Not a Complete Listing) (cont'd.)**

<table>
<thead>
<tr>
<th>Character</th>
<th>EBCDIC Code Point</th>
<th>ASCII Code Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>:</td>
<td>X'7A'</td>
<td>X'3A'</td>
</tr>
<tr>
<td>!</td>
<td>X'4F'</td>
<td>X'21'</td>
</tr>
<tr>
<td></td>
<td>for CPGID = 500</td>
<td></td>
</tr>
<tr>
<td></td>
<td>X'4F'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for CPGID = 893</td>
<td></td>
</tr>
<tr>
<td></td>
<td>X'5A'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for CPGID = 1303</td>
<td></td>
</tr>
<tr>
<td>&quot;</td>
<td>X'7F'</td>
<td>X'22'</td>
</tr>
<tr>
<td>&amp;</td>
<td>X'50'</td>
<td>X'26'</td>
</tr>
<tr>
<td>'</td>
<td>X'7D'</td>
<td>X'27'</td>
</tr>
<tr>
<td>(</td>
<td>X'4D'</td>
<td>X'28'</td>
</tr>
<tr>
<td>)</td>
<td>X'5D'</td>
<td>X'29'</td>
</tr>
<tr>
<td>[</td>
<td>X'4A'</td>
<td>X'5B'</td>
</tr>
<tr>
<td>*</td>
<td>X'5C'</td>
<td>X'2A'</td>
</tr>
<tr>
<td>,</td>
<td>X'6B'</td>
<td>X'2C'</td>
</tr>
<tr>
<td>;</td>
<td>X'5E'</td>
<td>X'3B'</td>
</tr>
<tr>
<td>&lt;</td>
<td>X'4C'</td>
<td>X'3C'</td>
</tr>
<tr>
<td>=</td>
<td>X'7E'</td>
<td>X'3D'</td>
</tr>
<tr>
<td>&gt;</td>
<td>X'6E'</td>
<td>X'3E'</td>
</tr>
<tr>
<td>?</td>
<td>X'6F'</td>
<td>X'3F'</td>
</tr>
<tr>
<td>–</td>
<td>X'6D'</td>
<td>X'5F'</td>
</tr>
<tr>
<td>Space</td>
<td>X'40'</td>
<td>X'20'</td>
</tr>
<tr>
<td>FNC1</td>
<td>X'8F'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for CPGID = 1303</td>
<td></td>
</tr>
<tr>
<td>RS (record separator)</td>
<td>X'1E'</td>
<td>X'1E'</td>
</tr>
<tr>
<td>GS (group separator)</td>
<td>X'1D'</td>
<td>X'1D'</td>
</tr>
<tr>
<td>US (unit separator)</td>
<td>X'1F'</td>
<td>X'1F'</td>
</tr>
<tr>
<td>EOT (end of transmission)</td>
<td>X'37'</td>
<td>X'04'</td>
</tr>
</tbody>
</table>
Code 128 Code Page

The Code 128 code page (CPGID = 1303, GCSGID = 1454) is defined as shown in Figure 14. This code page is used for all Code 128 symbols (Code 128, GS1-128, UCC/EAN 128, GIM USS-128) and GS1 DataBar symbols.

**Figure 14. Code 128 Code Page (CPGID = 1303, GCSGID = 1454)**

| Hex Digits | 0- | 1- | 2- | 3- | 4- | 5- | 6- | 7- | 8- | 9- | A- | B- | C- | D- | E- | F- |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|
|            | SE010000 | SE170000 | (SP) | & | _ | ^ | { | } | \ | 0 |
| -1         | SE020000 | SE180000 | / | LA010000 | j | ~ | A | J | 1 | ND10000 |
| -2         | SE030000 | SE190000 | FS | SE250000 | b | k | s | B | K | S | 2 | ND10000 |
| -3         | SE040000 | SE200000 | c | LL010000 | t | C | L | T | 3 | ND10000 |
| -4         | SE050000 | SE210000 | d | LR010000 | m | u | D | M | U | 4 | ND10000 |
| -5         | SE060000 | SE220000 | e | LE010000 | n | v | E | N | V | 5 | ND10000 |
| -6         | BS        | ETB      | f | LF010000 | o | w | F | O | W | 6 | ND10000 |
| -7         | SE270000 | SE380000 | g | LG010000 | p | x | G | P | X | 7 | ND10000 |
| -8         | SE280000 | SE390000 | h | LH010000 | q | y | H | Q | Y | 8 | ND10000 |
| -9         | ER        | SE400000 | i | LR010000 | r | z | I | R | Z | 9 | ND10000 |
| -A         | !         | SP020000 | ! | SD130000 | | | | | | | | | | | | |
| -B         | VT        | SE120000 | SP100000 | S | , | # | ] | | | | | | | | | | | |
| -C         | FF        | SE130000 | DC4 | SE210000 | | | | | | | | | | | | |
| -D         | CR        | SE140000 | ENQ | SE260000 | NAK | SE220000 | | | | | | | | | | | |
| -E         | SO        | SE150000 | ACK | SE270000 | | | | | | | | | | | | |
| -F         | SI        | SE160000 | US | SE380000 | | | | | | | | | | | | |

**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‘0F’ (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 FNC1, FNC2, FNC3, and FNC4 in the Code 128 Symbology Specification.
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.

Note: The BCOCA data-check and specification-check exception conditions are registered in the exception reporting chapter of the IPDS Reference. All new BCOCA exception conditions must also be registered in the IPDS Reference in addition to being defined in this chapter.

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.

<table>
<thead>
<tr>
<th>Exception</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC-0100</td>
<td>Retired item 4</td>
</tr>
<tr>
<td>EC-0200</td>
<td>Retired item 5</td>
</tr>
<tr>
<td>EC-0300</td>
<td>The bar code type specified in the BSD data structure is invalid or unsupported. Standard Action: Terminate bar code object processing.</td>
</tr>
<tr>
<td>EC-0400</td>
<td>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.</td>
</tr>
<tr>
<td>EC-0500</td>
<td>The color specified in the BSD data structure is invalid or unsupported. Standard Action: The device default color is used.</td>
</tr>
<tr>
<td>EC-0505</td>
<td>The unit base specified in the BSD data structure is invalid or unsupported. Standard Action: Terminate bar code object processing.</td>
</tr>
<tr>
<td>EC-0600</td>
<td>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.</td>
</tr>
<tr>
<td>EC-0605</td>
<td>The units per unit base specified in the BSD data structure is invalid or unsupported. Standard Action: Terminate bar code object processing.</td>
</tr>
<tr>
<td>EC-0610</td>
<td>The desired-symbol-width parameter value is invalid. Standard Action: Use a value of X'0000' for this parameter.</td>
</tr>
<tr>
<td>EC-0611</td>
<td>A desired symbol width was specified, but a bar code symbol cannot be generated that fits within the specified width. Standard Action: Use a value of X'0000' for the desired-symbol-width parameter; the resulting symbol is larger than the desired symbol width.</td>
</tr>
</tbody>
</table>
Specification-Check Exceptions

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-0805  The element height and height multiplier values specified are invalid for the selected GS1 DataBar modifier.
Standard Action: Use the smallest height defined for the GS1 DataBar modifier value.

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 (modified Plessey code) bar code, however, uses a default wide-to-narrow ratio of 2.00 to 1.

EC-0A00  The bar code origin (X offset value or Y offset 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-0D00  Retired item 6

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 (modified Plessey code) 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.
**Specifcation-Check Exceptions**

**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: Present the bar code symbol without structured append information.

**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 GS1 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 GS1 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 reader programming flag was set to B'1'.
- 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.
Specification-Check Exceptions

EC-0F0E  For a QR Code symbol, an invalid or unsupported conversion 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: Terminate bar code processing.

EC-1000  The human-readable interpretation location specified in 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, the Bearer Bars (Interleaved 2-of-5), the Identification Bars and USPS Banner (Intelligent Mail Container Barcode), the RED TAG indicator (Royal Mail RED TAG (deprecated)), the zipper pattern and contrast block (MaxiCode), 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 barcodes must be presented in their entirety. Whenever a partial bar code pattern is presented, for whatever reason, it is obscured to make it unscannable.

EC-1200  Invalid data was encountered in a GS1 DataBar Expanded, GS1 DataBar Expanded Stacked, GS1-128, or UCC/EAN 128 or symbol; one or more of the following conditions was encountered:
        • FNC1 is not the first data character (for UCC/EAN 128 symbols only)
        • Invalid application identifier (ai) value encountered
        • Data for an ai doesn't match the ai definition
        • Insufficient (or no) data following an ai
        • Too much data for an ai
        • Invalid use of FNC1 character
        Standard Action: Terminate bar code object processing.

EC-1201  Within a Data Matrix bar code object, a C40, Text, X12, or EDIFACT encodation scheme was selected and a character was encountered within the bar code data that is not valid for that encodation scheme. These encodation schemes do not support all 256 possible input characters.
        Standard Action: Produce the bar code symbol using the auto-encoding encodation scheme.
EC-1202  Invalid or insufficient data was encountered in a Royal Mail RED TAG (deprecated) bar code object. There must be exactly 21 numeric digits in the input data.
Standard Action: Terminate bar code object processing.

EC-1203  Invalid or insufficient data was encountered in an Intelligent Mail Container Barcode object. There must be exactly 22 characters in the input data that are within the field ranges shown in Table 15 on page 60.
Standard Action: Terminate bar code object processing.

EC-1204  Invalid, insufficient, or too much data was encountered in a Royal Mail Mailmark barcode. The valid data for the parameters of each modifier (X'00'–X'01') is defined within the Royal Mail Mailmark Definition Document.
Standard Action: Terminate bar code object processing.

Data-Check Exceptions

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

<table>
<thead>
<tr>
<th>Exception</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 29. For each bar code symbology type to be generated, one and only one BSD can be specified. For each BSD, zero or more BSAs can 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 values defined in the BCD1 or BCD2 range
- Detect and report to the controlling environment all exception conditions for supported values as defined in Chapter 5, "Exception Conditions", on page 131
- 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 139

A compliant receiver may in addition receive and process any BCOCA data structure value not in BCD1 or BCD2.
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.

Other sources can be found on the world wide web, one good example is the Barcode Software Center (http://www.makebarcode.com/info/info.html). This site also lists software packages, fonts, function libraries, printing hardware, books about bar codes, and worldwide organizations that maintain standards and specifications.

Bar code symbology specifications referred to in this book include:

• AIM International Technical Specification – International Symbology Specification
  – Data Matrix
  – MaxiCode
  – QR Code

• AIM Uniform Symbology Specification
  – Code 93
  – PDF417
  – USS-128 (also known as Code 128)
  – USS-Codabar (also known as Codabar)

  – Code 39
  – Interleaved 2 of 5
  – Code 11
  – Code 93
  – Code 49

• 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
  – Code 39 (3-of-9)
  – Codabar

  – Code 39

• Australia Post Bar Code; these publications are available from Australia Post:
  – Customer Barcoding Technical Specifications
  – A Guide to Printing the 4-State Barcode

  – MSI (modified Plessey code)
  – UPC/EAN
  – Code 39
  – Interleaved 2-of-5
Bar Code Symbology Specification References

- **Bar Code Specification by the Automotive Industry Action Group**, AIAG, Southfield, MI.
  - Code 39 (3-of-9)
  - Interleaved 2-of-5
- **Customer Guide to Confirm using PLANET Codes**, United States Postal Service
  - PLANET (deprecated)
  - EAN-8, EAN-13, Two-Digit Supplemental, Five-Digit Supplemental
- **GS1 General Specifications**, GS1 standards organization
  - UPC-A, UPC-E, Two-Digit Supplemental, Five-Digit Supplemental
  - EAN-8, EAN-13, Two-Digit Supplemental, Five-Digit Supplemental
  - ITF-14
  - GS1-128
  - GS1 DataBar
  - GS1 DataMatrix
- Intelligent Mail bar codes:
  - *Introducing 4-state Customer Barcode*, United States Postal Service
  - *Intelligent Mail Barcode (4-State Customer Barcode)*, United States Postal Service Specification (USPS-B-3200)
  - **BARCODE, CONTAINER, INTELLIGENT MAIL**, United States Postal Service Specification (USPS-B-3215)
  - Japan Postal Bar Code
  - A Japanese version of the specification is available online (www.post.japanpost.jp/zipcode/zipmanual/).
  - JAN-Short, -Standard
  - Code 39 (3-of-9)
- **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)
- Reduced Space Symbol bar codes (now called GS1 DataBar):
  - *AIM International Symbology Specification – Reduced Space Symbology (RSS)*
  - **ISO/IEC 24724 – GS1 DataBar bar code specification**
- Royal Mail RM4SCC bar codes:
  - *Royal Mail Customer Barcoding Trial Report & Technical Specification*
  - *Singapore Post, 4-State Bar Code for Customer Encoding*
- **KIX Technical Specifications**, PostNL (previously was the Mail division of TNT Post)
  - Dutch KIX
- **Royal Mail RED TAG Mailpiece Requirements**, Royal Mail Group Ltd.
  - Royal Mail RED TAG (deprecated)
- **Royal Mail Mailmark Definition Document**
  - Royal Mail Mailmark
  - Royal Mail Complex Mail Data Marks
Bar Code Symbology Specification References

- **UCC/EAN-128 Application Identifier Standard**, Uniform Code Council, Inc. Dayton, Ohio
  - UCC/EAN 128

- **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)

  - POSTNET (deprecated)

  - UPC-A, UPC-E, Two-Digit Supplemental, Five-Digit Supplemental
  - CGPC-A, CGPC-E
Appendix B. MO:DCA Environment

This appendix describes how bar code objects may be included within a MO:DCA 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 (MO:DCA) Reference for a full description of the MO:DCA architecture.

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

Bar Codes in MO:DCA Documents

The MO:DCA 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 bar code object area.

The MO:DCA 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 144.

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
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.

Table 31. MO:DCA Bar Code Data Descriptor (BDD)

<table>
<thead>
<tr>
<th>Structured Field Introducer</th>
<th>SF Length</th>
<th>SF Identifier</th>
<th>Flags</th>
<th>Reserved (2 bytes); should be X'0000'</th>
<th>Bar Code Symbol Descriptor followed by zero or one Color Specification (X'4E') triplets</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X'D3A6EB'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The data portion of the BDD structured field is defined in "Bar Code Symbol Descriptor (BSD)" on page 31. When a Color Specification (X'4E') triplet is present in the BDD, this triplet overrides the color value specified in BSD bytes 15-16.

Note: Support for the Color Specification (X'4E') triplet in the MO:DCA BDD structured field is part of the BCD2 subset of BCOCA.

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

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:

Table 32. MO:DCA Bar Code Data (BDA)

<table>
<thead>
<tr>
<th>Structured Field Introducer</th>
<th>SF Length</th>
<th>SF Identifier</th>
<th>Flags</th>
<th>Reserved (2 bytes); should be X'0000'</th>
<th>Bar Code Symbol Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X'D3EEEB'</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The data portion of the BDA structured field is described in "Bar Code Symbol Data (BSA)" on page 88.

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

Intelligent Printer Data Stream (IPDS) is the AFPC 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 or BCD2 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, the Bearer Bars (Interleaved 2-of-5), the Identification Bars and USPS Banner (Intelligent Mail Container Barcode), the RED TAG indicator (Royal Mail RED TAG
(deprecated)), the zipper pattern and contrast block (MaxiCode), 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:

TABLE 33. IPDS Bar Code Data Descriptor (BCDD)

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
<th>BCD2 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–1</td>
<td>UBIN</td>
<td>LENGTH</td>
<td>X'001B' – end of BCDD</td>
<td>Length of BCDD</td>
<td>X'001B' – end of BCDD</td>
<td>X'001B' – end of BCDD</td>
</tr>
<tr>
<td>2–3</td>
<td>CODE</td>
<td>SDF ID</td>
<td>X'A6EB'</td>
<td>BCDD Self-defining-field ID</td>
<td>X'A6EB'</td>
<td>X'A6EB'</td>
</tr>
<tr>
<td>27–end</td>
<td></td>
<td>Triplets</td>
<td>Zero or more optional triplets; not all IPDS printers support these triplets.</td>
<td>Triplets not supported in BCD1</td>
<td>Color Specification (X'4E') triplet</td>
<td></td>
</tr>
</tbody>
</table>

When a Color Specification (X'4E') triplet is present in the BCDD, this triplet overrides the color value specified in BSD bytes 15-16. If multiple X'4E' triplets are specified, the last one specified is used and the others are ignored. Printers that do not support extended bar code color support ignore all X'4E' triplets.

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 88.
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
BCOCA Exception Conditions and IPDS Exception IDs

The IPDS Architecture defines its own exception condition codes, called exception IDs, which consist of three bytes. BCOCA exception conditions are mapped to IPDS exception IDs by mapping the two-byte BCOCA code to the last two bytes of the IPDS exception ID; the first byte is either X'02', X'04', or X'08'. The IPDS Architecture also defines its own exception responses (called AEAs and PCAs). In some cases, this exception response is the same as the standard action defined by BCOCA. Where it is not, the IPDS exception response overrides the BCOCA standard action. Table 34 shows the mapping of BCOCA exception conditions to IPDS exception IDs.

Table 34. BCOCA Exception Conditions and IPDS Exception IDs

<table>
<thead>
<tr>
<th>BCOCA Exception Condition</th>
<th>IPDS Exception ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC-0300</td>
<td>X'0403..00'</td>
</tr>
<tr>
<td>EC-0400</td>
<td>X'0404..00'</td>
</tr>
<tr>
<td>EC-0500</td>
<td>X'0405..00'</td>
</tr>
<tr>
<td>EC-0505</td>
<td>X'0205..05'</td>
</tr>
<tr>
<td>EC-0600</td>
<td>X'0406..00'</td>
</tr>
<tr>
<td>EC-0605</td>
<td>X'0206..05'</td>
</tr>
<tr>
<td>EC-0610</td>
<td>X'0406..10'</td>
</tr>
<tr>
<td>EC-0611</td>
<td>X'0406..11'</td>
</tr>
<tr>
<td>EC-0700</td>
<td>X'0407..00'</td>
</tr>
<tr>
<td>EC-0705</td>
<td>X'0207..05'</td>
</tr>
<tr>
<td>EC-0800</td>
<td>X'0408..00'</td>
</tr>
<tr>
<td>EC-0805</td>
<td>X'0408..05'</td>
</tr>
<tr>
<td>EC-0900</td>
<td>X'0409..00'</td>
</tr>
<tr>
<td>EC-0A00</td>
<td>X'040A..00'</td>
</tr>
<tr>
<td>EC-0B00</td>
<td>X'040B..00'</td>
</tr>
<tr>
<td>EC-0C00</td>
<td>X'040C..00'</td>
</tr>
<tr>
<td>EC-0E00</td>
<td>X'040E..00'</td>
</tr>
<tr>
<td>EC-0F00</td>
<td>X'040F..00'</td>
</tr>
<tr>
<td>EC-0F01</td>
<td>X'040F..01'</td>
</tr>
<tr>
<td>EC-0F02</td>
<td>X'040F..02'</td>
</tr>
<tr>
<td>EC-0F03</td>
<td>X'040F..03'</td>
</tr>
<tr>
<td>EC-0F04</td>
<td>X'040F..04'</td>
</tr>
<tr>
<td>EC-0F05</td>
<td>X'040F..05'</td>
</tr>
<tr>
<td>EC-0F06</td>
<td>X'040F..06'</td>
</tr>
<tr>
<td>EC-0F07</td>
<td>X'040F..07'</td>
</tr>
<tr>
<td>EC-0F08</td>
<td>X'040F..08'</td>
</tr>
<tr>
<td>EC-0F09</td>
<td>X'040F..09'</td>
</tr>
<tr>
<td>EC-0F0A</td>
<td>X'040F..0A'</td>
</tr>
<tr>
<td>EC-0F0B</td>
<td>X'040F..0B'</td>
</tr>
<tr>
<td>EC-0F0C</td>
<td>X'040F..0C'</td>
</tr>
</tbody>
</table>
### Table 34  BCOCA Exception Conditions and IPDS Exception IDs (cont’d.)

<table>
<thead>
<tr>
<th>BCOCA Exception Condition</th>
<th>IPDS Exception ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC-0F0D</td>
<td>X'040F..0D'</td>
</tr>
<tr>
<td>EC-0F0E</td>
<td>X'040F..0E'</td>
</tr>
<tr>
<td>EC-0F0F</td>
<td>X'040F..0F'</td>
</tr>
<tr>
<td>EC-0F10</td>
<td>X'040F..10'</td>
</tr>
<tr>
<td>EC-0F11</td>
<td>X'040F..11'</td>
</tr>
<tr>
<td>EC-0F12</td>
<td>X'040F..12'</td>
</tr>
<tr>
<td>EC-1000</td>
<td>X'0410..00'</td>
</tr>
<tr>
<td>EC-1100</td>
<td>X'0411..00'</td>
</tr>
<tr>
<td>EC-1200</td>
<td>X'0412..00'</td>
</tr>
<tr>
<td>EC-1201</td>
<td>X'0412..01'</td>
</tr>
<tr>
<td>EC-1202</td>
<td>X'0412..02'</td>
</tr>
<tr>
<td>EC-1203</td>
<td>X'0412..03'</td>
</tr>
<tr>
<td>EC-1204</td>
<td>X'0412..04'</td>
</tr>
<tr>
<td>EC-2100</td>
<td>X'0821..00'</td>
</tr>
</tbody>
</table>
Appendix D. Retired Items

This appendix lists each retired item that is mentioned within the body of this book and also lists those items that have been unretired.

**Retired item 1 (1991):** X'02' in the unit base field (byte 0) of the Bar Code Symbol Descriptor (BSD) structure is retired for relative units.

**Retired item 2 (1991):** This retired item was *unretired in 1993*; note that this bit was retired again with retired item 21.

Byte 0, bit 4 of the Bar Code Symbol Data (BSA) structure is retired for IBM PC ASCII data stream use; in particular this flag is used by the IBM Personal Printer Data Stream (PPDS) to indicate ASCII data.

**Retired item 3 (1991):** Byte 0, bit 7 of the Bar Code Symbol Data (BSA) structure is retired for IBM PC ASCII data stream use; in particular this flag is used by the IBM Personal Printer Data Stream (PPDS).

**Retired item 4 (1991):** Exception Code EC-0100 is retired for IBM channel-attached printers (used at the 370-channel protocol level to indicate that a channel overrun has occurred).

**Retired item 5 (1991):** Exception Code EC-0200 is retired for IBM 4224 and 4234 printers (attempt to print symbol or HRI character out of object area).

**Retired item 6 (1991):** Exception Code EC-0D00 is retired for IBM 4224 and 4234 printers (symbol reference point outside logical page).

**Retired item 7 (1992):** Bar Code Symbol Descriptor type (byte 12) X'04' – UPC/CGPC - Version D, modifiers (byte 13) X'00' through X'04' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>Present a UPC Version D-1 (Block-1) bar code with a generated check digit. Block-1 contains thirteen data characters and the check digit.</td>
</tr>
<tr>
<td>X'01'</td>
<td>Present a UPC Version D-2 (Block-2 + Block-3) bar code with two generated check digits, one for each block. Blocks-2 and -3 contain eighteen data characters and the two check digits.</td>
</tr>
<tr>
<td>X'02'</td>
<td>Present a UPC Version D-3 (Block-2 + Block-6) bar code with two generated check digits, one for each block. Blocks-2 and -6 contain twenty-two data characters and the two check digits.</td>
</tr>
<tr>
<td>X'03'</td>
<td>Present a UPC Version D-4 (Block-2 + Block-4 + Block-5) bar code with three generated check digits, one for each block. Blocks-2, -4 and -5 contain twenty-five data characters and the three check digits.</td>
</tr>
<tr>
<td>X'04'</td>
<td>Present a UPC Version D-5 (Block-2 + Block-5 + Block-7) bar code with three generated check digits, one for each block. Blocks-2, -5 and -7 contain twenty-nine data characters and the three check digits.</td>
</tr>
</tbody>
</table>
Retired Items

Retired item 8 (1992): This retired item was **unretired in 1993**.

This item was previously retired for Bar Code Symbol Descriptor type (byte 12) X'06' – UPC-Two-digit Supplemental (Periodicals), modifiers (byte 13) X'01' through X'02'.

Retired item 9 (1992): This retired item was **unretired in 1993**.

This item was previously retired for Bar Code Symbol Descriptor type (byte 12) X'07' – UPC-Five-digit Supplemental (Paperbacks), modifiers (byte 13) X'01' through X'02'.

Retired item 10 (1992): Bar Code Symbol Descriptor type (byte 12) X'0E' – Jan Short, modifier (byte 13) X'00' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>Present a JAN Short 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.</td>
</tr>
</tbody>
</table>

Retired item 11 (1992): Bar Code Symbol Descriptor type (byte 12) X'0F' – Jan Standard, modifier (byte 13) X'00' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>Present a JAN Standard 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 the generated check digit are encoded. The first flag digit is presented in human readable form at the bottom of the left quiet zone. The first flag digit also governs the A or B number set pattern of the bar and space pattern of the six digits to the left of the symbol center pattern.</td>
</tr>
</tbody>
</table>

Retired item 12 (1992): Bar Code Symbol Descriptor type (byte 12) X'10' – Subset of 3-of-9 Code, MHI/AIM USD-2, modifiers (byte 13) X'01' through X'02' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Present the bar code symbol without a generated check digit.</td>
</tr>
<tr>
<td>X'02'</td>
<td>Generate a check digit and present it with the bar code.</td>
</tr>
</tbody>
</table>

Retired item 13 (1992): Bar Code Symbol Descriptor type (byte 12) X'12' – Code 93, AIM USS-93 (ASCII not included), modifiers (byte 13) X'01' through X'02' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Present the bar code symbol without a generated check digit.</td>
</tr>
<tr>
<td>X'02'</td>
<td>Generate two check digits and present them with the bar code.</td>
</tr>
</tbody>
</table>
**Retired item 14 (1992):** Bar Code Symbol Descriptor type (byte 12) X'13' – Code 11, MHI/AIM USD-8, modifiers (byte 13) X'01' through X'03' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Present the bar code symbol without a generated check digit.</td>
</tr>
<tr>
<td>X'02'</td>
<td>Generate two check digits and present them with the bar code.</td>
</tr>
<tr>
<td>X'03'</td>
<td>Generate a check digit and present it with the bar code.</td>
</tr>
</tbody>
</table>

**Retired item 15 (1992):** Bar Code Symbol Descriptor type (byte 12) X'14' – ASCII Version, Code 93, AIM USS-93, modifier (byte 13) X'00' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>Present the bar code symbol without a generated check digit.</td>
</tr>
</tbody>
</table>

**Retired item 16 (1992):** Bar Code Symbol Descriptor type (byte 12) X'15' – Plessey, modifiers (byte 13) X'01' through X'02' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Present the bar code symbol without a generated check digit.</td>
</tr>
<tr>
<td>X'02'</td>
<td>Generate a check digit and present it with the bar code.</td>
</tr>
</tbody>
</table>

**Retired item 17 (1992):** Bar Code Symbol Descriptor type (byte 12) X'16' – EAN Two-Digit Supplemental, modifiers (byte 13) X'02' through X'03' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'02'</td>
<td>The two-digit supplemental bar code symbol is preceded by a EAN-13 bar code symbol. The bar code object contains both the EAN-13 symbol and the two-digit supplemental symbol. The input data consists of three flags digits, a nine-digit ISBN (International Standard Book Numbering) number, and the two supplement digits, in that order. A check digit is generated for the pseudo EAN-13 symbol using the regular EAN algorithm. 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. <strong>Note:</strong> Restricted to books and paperbacks.</td>
</tr>
<tr>
<td>X'03'</td>
<td>Reserved for future periodical use.</td>
</tr>
</tbody>
</table>
Retired Items

Retired item 18 (1992): Bar Code Symbol Descriptor type (byte 12) X'17' – EAN Five-Digit Supplemental, modifiers (byte 13) X'02' through X'03' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'02'</td>
<td>The five-digit supplemental bar code symbol is preceded by a EAN-13 bar code symbol. The bar code object contains both the EAN-13 symbol and the five-digit supplemental symbol. The input data consists of three flags digits, a nine-digit ISBN (International Standard Book Numbering) number, and the five supplement digits, in that order. A check digit is generated for the pseudo EAN-13 symbol using the regular EAN algorithm. A second check digit is generated from the five-digit supplemental data. The second check digit is used to assign the bar and space patterns for the supplemental data from number sets A and B. The second check digit is not encoded or interpreted.</td>
</tr>
<tr>
<td>X'03'</td>
<td>Reserved for future periodical use.</td>
</tr>
</tbody>
</table>

Note: Restricted to books and paperbacks.

Retired item 19 (1992): Bar Code Symbol Descriptor type (byte 12) X'19' – Facing Identification Mark (FIM) - United States Postal Service, modifiers (byte 13) X'00' through X'03' is retired with the following meaning:

For all FIM 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 FIM-A; these fields are ignored. The FIM Specification defines specific values for these parameters.

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'00'</td>
<td>Present a FIM Type A bar code symbol. FIM-A is used for courtesy reply mail that also uses the POSTNET bar code. There is no data variable to the Bar Code Symbol Data (BSA) data structure. The user is required to present the permit holder's complete address (company name, street address, city, state, and nine-digit, ZIP+4 code) and the POSTNET ZIP+4 bar code as separate presentation spaces.</td>
</tr>
</tbody>
</table>
| X'01'    | Present a FIM Type B bar code symbol. FIM-B is used for business reply mail, penalty, or franked mail that does not use the POSTNET bar code. There is no data variable to the Bar Code Symbol Data (BSA) data structure. The user is required to present:
  ◦ The permit holder's complete address (company name, street address, city, state, and nine-digit, ZIP+4 code)
  ◦ The "No Postage Necessary If Mailed In The United States" endorsement
  ◦ The business reply legend: 1) "Business Reply Mail", 2) permit number – “First-Class Mail Permit No XXXXX” (five digits), and 3) city and state where the permit was granted
  ◦ The postage paid endorsement “Postage Will Be Paid By Addressee”
  ◦ The series of horizontal bars under the “No Postage Necessary ...” endorsement as separate presentation spaces. |
| X'02'    | Present a FIM Type C bar code symbol. FIM-C is used for business reply mail, penalty, or franked mail that also uses the POSTNET bar code. There is no data variable to the Bar Code Symbol Data (BSA) data structure. The user is required to present:
  ◦ The permit holder's complete address (company name, street address, city, state, and nine-digit, ZIP+4 code)
  ◦ The "No Postage Necessary If Mailed In The United States" endorsement |
Retired Items

- The business reply legend: 1) "Business Reply Mail", 2) permit number – “First-Class Mail Permit No XXXXX” (five digits), and 3) city and state where the permit was granted.
- The postage paid endorsement “Postage Will Be Paid By Addressee”
- The series of horizontal bars under the "No Postage Necessary ..."
- Permit holder's POSTNET ZIP+4 bar code as separate presentation spaces.

X'03' Present a FIM Type D bar code symbol. FIM-D is used for OCR readable mail (usually used on courtesy reply window envelopes) that does not use the POSTNET bar code. There is no data variable in the Bar Code Symbol Data (BSA) data structure for FIM-D.

The user is required to present the permit holder’s complete address (company name, street address, city, state) and nine-digit, ZIP+4 code as separate presentation spaces. The ZIP+4 code is obtained from the address information when scanned by an OCR reader.

Retired item 20 (1993): Bar Code Symbol Descriptor type (byte 12) X'11' – Code 128, AIM USS-128, modifier (byte 13) X'01' is retired with the following meaning:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'01'</td>
<td>Present the bar code symbol without a generated check digit.</td>
</tr>
</tbody>
</table>

Retired item 21 (2006): Byte 0, bit 4 of the Bar Code Symbol Data (BSA) structure is retired because it was not used in a product; in particular this flag is intended to allow for both ASCII and EBCDIC data, but only EBCDIC data is actually used.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Type</th>
<th>Name</th>
<th>Range</th>
<th>Meaning</th>
<th>BCD1 Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>BITS</td>
<td>Flags</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 4</td>
<td>CPTYPE</td>
<td>CPtype</td>
<td>B'0' B'1'</td>
<td>Code page type: EBCDIC-based ASCII-based</td>
<td>B'0'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Note: Not all BCOCA receivers support ASCII-based code pages.</td>
<td></td>
</tr>
</tbody>
</table>

Bit 4 CPTYPE

This flag specifies the type of code page used to encode the data field; the choices are shown in Table 35 on page 156. For the Code 128, Data Matrix, MaxiCode, PDF417, and QR Code symbologies, this flag is ignored.

Support for the EBCDIC-based code pages is mandatory, but support for the ASCII-based code pages is optional. BCOCA receivers that support only the mandatory code pages will ignore this flag, and the bar code symbol will be presented incorrectly if the data was encoded using an ASCII-based code page. Refer to the product documentation for your BCOCA receiver product (such as a printer) to determine which type of code pages are supported.

If bit 4 is B'0', an EBCDIC-based code page is used.

If bit 4 is B'1', an ASCII-based code page is used.
### Table 35. Valid Code Pages and Type Styles

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>EBCDIC-Based CPGID</th>
<th>ASCII-Based CPGID</th>
<th>FGID</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'02'</td>
<td>MSI (modified Plessey code)</td>
<td>500</td>
<td>850</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'03'</td>
<td>UPC/CGPC — Version A</td>
<td>893</td>
<td>850</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'05'</td>
<td>UPC/CGPC — Version E</td>
<td>893</td>
<td>850</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'06'</td>
<td>UPC — Two-digit Supplemental (Periodicals)</td>
<td>893</td>
<td>850</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'07'</td>
<td>UPC — Five-digit Supplemental (Paperbacks)</td>
<td>893</td>
<td>850</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'08'</td>
<td>EAN-8 (includes JAN-short)</td>
<td>893</td>
<td>850</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'09'</td>
<td>EAN-13 (includes JAN-standard)</td>
<td>893</td>
<td>850</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'0A'</td>
<td>Industrial 2-of-5</td>
<td>500</td>
<td>850</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'0B'</td>
<td>Matrix 2-of-5</td>
<td>500</td>
<td>850</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'0C'</td>
<td>Interleaved 2-of-5, AIM USS-1 2/5</td>
<td>500</td>
<td>850</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'0D'</td>
<td>Codabar, 2-of-7, AIM USS-Codabar</td>
<td>500</td>
<td>850</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'11'</td>
<td>Code 128, AIM USS-128</td>
<td>1303</td>
<td>None</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'16'</td>
<td>EAN Two-digit Supplemental</td>
<td>893</td>
<td>850</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'17'</td>
<td>EAN Five-digit Supplemental</td>
<td>893</td>
<td>850</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>X'18'</td>
<td>POSTNET</td>
<td>500</td>
<td>850</td>
<td>None</td>
</tr>
<tr>
<td>X'1A'</td>
<td>RM4SCC</td>
<td>500</td>
<td>850</td>
<td>None</td>
</tr>
<tr>
<td>X'1B'</td>
<td>Japan Postal Bar Code</td>
<td>500</td>
<td>850</td>
<td>None</td>
</tr>
<tr>
<td>X'1C'</td>
<td>Data Matrix (2D bar code)</td>
<td></td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>X'1D'</td>
<td>MaxiCode (2D bar code)</td>
<td></td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>X'1E'</td>
<td>PDF417 (2D bar code)</td>
<td></td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>X'1F'</td>
<td>Australia Post Bar Code</td>
<td>500</td>
<td>850</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'20'</td>
<td>QR Code</td>
<td></td>
<td></td>
<td>None</td>
</tr>
<tr>
<td>X'21'</td>
<td>Code 93</td>
<td>500</td>
<td>850</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'22'</td>
<td>USPS Four-State</td>
<td>500</td>
<td>850</td>
<td>Device specific</td>
</tr>
<tr>
<td>X'23'</td>
<td>Reduced Space Symbology (RSS)</td>
<td>1303</td>
<td>None</td>
<td>Device specific</td>
</tr>
</tbody>
</table>
Retired item 22 (2011): Bar Code Symbol Descriptor type (byte 12) X'EC', modifier (byte 13) X'02' is retired as Océ private-use values to indicate QR Code.

Retired item 23 (2011): Bar Code Symbol Descriptor type (byte 12) X'ED', modifier (byte 13) X'00' is retired as Océ private-use values to indicate Maxicode.

Retired item 24 (2011): Bar Code Symbol Descriptor type (byte 12) X'EE', modifier (byte 13) X'00' is retired as Océ private-use values to indicate Data Matrix.

Retired item 25 (2011): Bar Code Symbol Descriptor type (byte 12) X'EF', modifiers (byte 13) X'00' through X'01' is retired as Océ private-use values to indicate PDF 417.
Notices

The AFP Consortium or consortium member companies might have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents.

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

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. The AFP Consortium might make improvements and/or changes in the architecture described in this publication at any time without notice.

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

The AFP Consortium may use or distribute any information you supply in any way it believes appropriate without incurring any obligation to you.

This information contains examples of data and reports used in daily business operations. To illustrate them in a complete manner, some examples include the names of individuals, companies, brands, or products. These names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.
Trademarks

These terms are trademarks or registered trademarks of Ricoh Co., Ltd., in the United States, other countries, or both:

ACMA™
Advanced Function Presentation™
AFP™
AFP Color Consortium™
AFP Color Management Architecture™
AFPCC™
Bar Code Object Content Architecture™
BCOCA™
CMOCA™
Color Management Object Content Architecture™
InfoPrint®
Infoprint®
Intelligent Printer Data Stream™
IPDS™
Mixed Object Document Content Architecture™
MO:DCA™
Ricoh®

Adobe®, the Adobe logo, PostScript®, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

AFPC™ and AFP Consortium™ are trademarks of the AFP Consortium.

IBM® is a registered trademark of the International Business Machines Corporation in the United States, other countries, or both. MVS™ is a trademark of the International Business Machines Corporation in the United States, other countries, or both.

Intelligent Mail® is a registered trademark of the United States Postal Service.

Royal Mail Mailmark® is a registered trademark of Royal Mail Group Limited. Mailmark™ is a trademark of Royal Mail Group Limited.

QR Code® is a registered trademark of DENSO WAVE INCORPORATED.

UP3I™ is a trademark of UP3I Limited.

Other company, product, or service names may be 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 InfoPrint Dictionary of Printing.

The following definitions are provided as supporting information only, and are not intended to be used as a substitute for the semantics described in the body of this reference.

A

absolute coordinate. One of the coordinates that identify the location of an addressable point with respect to the origin of a specified coordinate system. Contrast with relative coordinate.

absolute move. A method used to designate a new presentation position by specifying the distance from the designated axes to the new presentation position. The reference for locating the new presentation position is a fixed position as opposed to the current presentation position.

absolute positioning. The establishment of a position within a coordinate system as an offset from the coordinate system origin. Contrast with relative positioning.

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 acknowledgment-required 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.

additive primary colors. Red, green, and blue light, transmitted in video monitors and televisions. When used in various degrees of intensity and variation, they create all other colors of light; when superimposed equally, they create white. Contrast with subtractive primary colors.

addressable position. A position in a presentation space or on a physical medium that can be identified by a coordinate from the coordinate system of the presentation space or physical medium. See also picture element. Synonymous with position.

Advanced Function Presentation (AFP). An open architecture for the management of presentable information that is developed by the AFP Consortium (AFPC). AFP comprises a number of data stream and data object architectures:

- Mixed Object Document Content (MO:DCA) Architecture; formerly referred to as AFPDS
- Intelligent Printer Data Stream (IPDS) Architecture
- AFP Line Data Architecture
- Bar Code Object Content Architecture (BCOCA)
- Color Management Object Content Architecture (CMOCA)
- Font Object Content Architecture (FOCA)
- Graphics Object Content Architecture for AFP (AFP GOCA)
- Image Object Content Architecture (IOCA)
- Metadata Object Content Architecture (MOCA)
- Presentation Text Object Content Architecture (PTOCA)

AEA. See alternate exception action.

AFM file. A file containing the metric information required for positioning the characters of a font. The metric information contained in this file was extracted from a PFB file, in an ASCII file format defined by Adobe Systems Inc., and used for character positioning and page formatting.

AFP. See Advanced Function Presentation.

AFP Consortium (AFPC). A formal open standards body that develops and maintains AFP architecture. Information about the consortium can be found at www.afpcinc.org.

AFP data stream. A presentation data stream that is processed in AFP environments. The MO:DCA architecture defines the strategic AFP interchange data stream. The IPDS architecture defines the strategic AFP printer data stream.

AFPD. 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.
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) 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.

article. The physical item that a bar code identifies.

ascender. The parts of certain lowercase letters, such as b, d, or f, that 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'), function no longer achievable (action code X'06'), 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.
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.

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 towers. 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 architectured 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
**baseline direction (B)** • **B-space**

**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 bounded character 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 (Bₒ)**. 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 axis of its bounding Xₒ,Yₒ 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.

**Bearer Bars**. Bars that surround an Interleaved 2-of-5 bar code to prevent misreads and short scans that might occur when a skewed scanning beam enters or exits the bar code symbol through its top or bottom edge. When plates are used in the printing process, Bearer Bars help equalize the pressure exerted by the printing plate over the entire surface of the symbol to improve print quality. There are two styles: 1) four bars that completely surround the bar/code pattern and 2) two bars that are placed at the top and bottom of the bar/code pattern.

**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 Yₑ-extent when the B axis is parallel to the Y axis or to the Xₑ-extent when the B axis is parallel to the X axis.

**bilevel custom pattern**. In GOCA, a custom pattern that is defined as uncolored, then has a single color assigned to it when it is used to fill an area. Contrast with full-color custom pattern.

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

**blend**. A mixing rule in which the intersection of part of a new presentation space $P_{\text{new}}$ with part of an existing presentation space $P_{\text{existing}}$ changes to a new color attribute that represents a color-mixing of the color attributes of $P_{\text{new}}$ with the color attributes of $P_{\text{existing}}$. For example, if $P_{\text{new}}$ has foreground color attribute blue and $P_{\text{existing}}$ has foreground color attribute yellow, the area where the two foregrounds intersect changes to a color attribute of green. See also **mixing rule**. Contrast with overpaint and underpaint.

**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.

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.

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-addressable. Allowing characters to be addressed, referenced, and positioned only in a fixed number of character-size rectangles into which a presentation space is divided. Contrast with all-points-addressable.

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 character shape and character 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 characters 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.

CID file. A file containing the font information required for presenting the characters of a font. The shape information (glyph procedures) contained in this file is in a binary encoded format defined by Adobe Systems Inc., optimized for large character set fonts (for example, Japanese ideographic fonts having several thousand characters).

CIE. See Commission Internationale d'Éclairage.

CIELAB color space. Internationally accepted color space model used as a standard to define color within the graphic arts industry, as well as other industries. L*, a*, and b* are plotted at right angles to one another. Equal distances in the space represent approximately equal color difference.

CIEXYZ color space. The fundamental CIE-based color space that allows colors to be expressed as a mixture of the three tristimulus values X, Y, and Z.

CJK fonts. Fonts that contain a set of unified ideographic characters used in the written Chinese, Japanese, and Korean languages. The character encoding is the same for each language, but there might be glyph variants between languages.

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.

CMAP file. A file containing the mapping of code points to the character index values used in a CID file. The code points conform to a particular character coding system which is used to identify the characters in a document data stream. The character index values are assigned in a CID file for identification of the glyph procedure used to define the character shape. The mapping information in this file is in an ASCII file format defined by Adobe Systems Inc.

CMOCA. See Color Management Object Content Architecture.

CMR. See Color management resource.

CMY. Cyan, magenta, and yellow, the subtractive primary colors.
CMYK color space. The color model used in four-color printing. Cyan, magenta, and yellow, the subtractive primary colors, are used with black to effectively create a multitude of other colors.

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

CODE. A data type for architecture syntax that indicates an architectural 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 controlling 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 GCSSID 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 GCSSID 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 page. See also code page and code point.

color. A visual attribute of things that results from the light they emit, transmit, or reflect.

color attribute. An attribute that affects the color values provided in a graphics primitive, a text control sequence, or an IPDS command. Examples of color attributes are foreground color and background color.

color component. A dimension of a color value expressed as a numeric value. For example, a color value might consist of one, two, three, four, or eight components, also referred to as channels.

color conversion. The process of converting colors from one color space to another.

color image. Images whose image data elements are represented by multiple bits or whose image data element values are mapped to color values. Constructs that map image-data-element values to color values are look-up tables and image-data-element structure parameters. Examples of color values are screen color values for displays and color toner values for printers.

color management. The technology to calibrate the color of input devices (such as scanners or digital cameras), display devices, and output devices (such as printers or offset presses).
**Color Management Object Content Architecture (CMOCA)**. An architectured collection of constructs used for the interchange and presentation of the color management information required to render a print file, document, group of pages or sheets, page, overlay, or data object with color fidelity.

**color management resource.** An object that provides color management in presentation environments.

**color management system.** A set of software designed to increase the accuracy and consistency of color between color devices like a scanner, display, and printer.

**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.

**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. The command introducer is environment dependent. (3) 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.

**Commission Internationale d’Eclairage (CIE).** An association of international color scientists who produced the standards that are used as the basis of the description of color.

**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 architectured 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 parameters.

**control sequence chaining.** A method used to identify a sequential string of control sequences so they can be processed efficiently.

**control sequence class.** An assigned coded character that identifies a control sequence’s syntax and how that syntax is to be interpreted. An example of a control sequence class is X’D3’, that 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 set.** A collection of pages intended to be printed multiple times. For example, when multiple copies of a book or booklet is printed, each copy of the book or booklet is a copy set. This term was originally used with copy machines to identify collections of copies that are delivered as sets or stapled as sets. The term was also used when printing multiple copies of an MVS™ data set.

**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.** 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*.

**CPI.** Characters per inch.

**C-space.** The distance from the most positive *character coordinate system* X-axis value of a *character shape* to the *character escapement point*. C-space can be positive, zero, or negative. See also *A-space* and *B-space*.

**current baseline coordinate.** The *baseline presentation position* at the present time. The baseline presentation position is the summation of the increments of all baseline controls since the baseline was established in the *presentation space*. The baseline presentation position is established in a presentation space either as part of the initialization procedures for processing an *object* or by an Absolute Move Baseline *control sequence*. Synonymous with *current baseline presentation coordinate*.

**current baseline presentation coordinate (B_c).** 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 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.

custom line type value. A user-defined line type, defined by a string of pairs of a dash/dot length followed by a move length. Contrast with standard line type value.

custom pattern. In GOCA, a user-defined pattern, defined by the picture drawn by a series of drawing orders between a Begin Custom Pattern drawing order and an End Custom Pattern drawing order. Custom patterns can be either bilevel custom patterns or full-color custom patterns. Contrast with patterns in the default pattern set.

custom pattern mode. In GOCA, a mode that is entered when a Begin Custom Pattern drawing order is executed and exited when an End Custom Pattern drawing order is executed. While in this mode, drawing is done in a separate, temporary graphics presentation space rather than in the graphics presentation space of the current GOCA object.


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. In the IPDS architecture, a presentation-form object that is either specified within a page or overlay or is activated as a resource and later included in a page or overlay via the IDO command. Examples include: PDF single-page objects, Encapsulated PostScript objects, and IO Images. See also resource and data object resource.

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 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 object resource. In the IPDS architecture, an object-container resource or IO-Image resource that is either printer resident or downloaded. Data object resources can be:

- Used to prepare for the presentation of a data object; such as with a Color Management Resource or Resident Color Profile Resource
- Included in a page or overlay via the Include Data Object command; examples include: PDF single-page objects, Encapsulated PostScript objects, and IO Images
- Invoked from within a data object; examples include: PDF Resource objects

See also data object and resource.

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.

default pattern set. In GOCA, a set of predefined patterns, like solid, dots, or horizontal lines. Contrast with custom pattern.

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.

deprecated. An architectured construct is marked as "deprecated" to indicate that it should no longer be used because it has been superseded by a newer construct. Use or support of a deprecated construct is permitted but no longer recommended. Constructs are deprecated rather than immediately removed to provide backward compatibility.

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.

define • digital

define. The semantic name or identifier of a primitive type or object. See also object id.

definition. A statement that describes or explains something.

definitions. A list of commonly used terms and their meanings.

definition symbol. An identifier that is not an object but that is used as a name in a definition.

definition symbol table. A collection of definition symbols and their meanings.

definition symbol table entry. An entry in a definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

definition symbol table manager. A software component responsible for managing the definition symbol table.

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 independent. Not dependent upon device characteristics.

device-independent color space. A CIE-based color space that allows color to be expressed in a device-independent way. It ensures colors to be predictably and accurately matched among various color devices.

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 profile. A structure that provides a means of defining the color characteristics of a given device in a particular state.

device resolution. The number of pels that can be printed in an inch, both horizontally and vertically. This is the resolution that the printer uses when printing. Some printers can be configured to print with a variety of resolutions that can be selected by the operator. The device resolution can be different in the two directions (for example, a resolution of 360 by 720).

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 bi-level 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.

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.

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, that can have a content portion that provides control information, data, or both. An application or device does not have to understand control information or data to parse a data stream when all the records in the data stream are document elements. See also structured field.

document fidelity. The degree to which a document presentation preserves the creator's intent.

document formatting. A method used to determine where information is positioned in presentation spaces or on physical media.

document presentation. A method used to produce a visible copy of formatted information on physical media.

double-byte character set (DBCS). A character set that can contain up to 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, 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.

duplex. A method used to print data on both sides of a sheet. Normal-duplex printing occurs when the sheet is turned over the Ym axis. Tumble-duplex printing occurs when the sheet is turned over the Xm 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.

Efficient XML Interchange (EXI). A format that allows XML documents to be encoded as binary data, rather than as plain text.

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) 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.

escalation 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. (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.

EXI. See Efficient XML 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.

Extensible Markup Language (XML). A set of rules for encoding documents in a format that is both human-readable and machine-readable.

Extensible Metadata Platform (XMP). An ISO standard, originally created by Adobe Systems Incorporated, for the creation, processing, and interchange of standardized and custom metadata for all kinds of resources.

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.

FGID. See Font Typeface Global Identifier.

Filename Map File. A file containing the mapping of object names to file names for use in establishing a font file system. Object names and file names do not conform to the same naming requirements, so it is necessary to provide a mapping between them. The mapping information in this file is in an ASCII file format defined by Adobe Systems Inc.

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.
fixed metrics • font production

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 barcode data provided within the data stream. Fixed medium information can be used to create preprinted 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.

FNN Linked. In FOCA, the FNN (Font Name Map) structured field permits the mapping of a set of IBM GCGIDs to the character index values which occur in either a CMAP file or a Rearranged file. Because the set of GCGIDs and the set of character index values must correspond to the same set of characters, it is necessary to identify which CMAP or Rearranged file (among the many that could be located in a font file system) is associated (linked) with the FNN structured field. Note that the Font Name Map is known as the Character ID Map in IPDS.

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 character shape, graphic pattern, style, size, weight class, 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 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 class, and width class. (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 local identifier. A binary identifier that is mapped by the controlling environment to a named resource to identify a font. See also local identifier.

font metrics. Measurement information that defines individual character values such as height, width, and space, as well as overall font values such as averages and maximums. Font metrics can be expressed in specific fixed units, such as pels, or in relative units that are independent of both the resolution and the size of the font. See also character metrics and character set metrics.

font modification parameters. Parameters that alter the appearance of a typeface.

font object. A resource object that contains some or all of the description of a font.

Font Object Content Architecture (FOCA). An architected collection of constructs used to describe fonts and to interchange those font descriptions.

font production. A method used to create a font. This method includes designing each character image, converting the character images to a digital-technology format, defining parameter values for each character, assigning appropriate descriptive and identifying information, and creating a font resource that contains the required information in a format that can be used by a text processing system. Digital-technology formats include bit image, vector drawing orders, and outline algorithms. Parameter values include such attributes as height, width, and escapement.
font referencing. A method used to identify or characterize a font. Examples of processes that use font referencing are document editing, document formatting, and document presentation.

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 class, and width class.

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, that 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, that 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 graphics 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-sheet media, continuous-form media, 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.

formatter. A process used to prepare a document for presentation.

formblend. This mixing rule is only used when a preprinted form overlay (PFO) is merged as presentation space P_{PFO} with other presentation data (presentation space P_{data}). The intersection of P_{PFO} and P_{data} is assigned the following color attribute:
- Wherever the color attribute of P_{PFO} is either color of medium, or “white” (CMYK = X’00000000’ for a printer, RGB = X’FFFFFF’ for an RGB display), the intersection is assigned the color attribute of P_{data}. Likewise, wherever the color attribute of P_{data} is either color of medium, or “white” (CMYK = X’00000000’ for a printer, RGB = X’FFFFFF’ for an RGB display), the intersection is assigned the color attribute of P_{PFO}.
- With other overlapping color values, the intersection assumes a new color attribute that is generated in a device-specific manner to simulate how the P_{data} color attribute would mix onto a preprinted form that has the color attribute of P_{PFO}. In general, this mixing is a blending of the color attributes of P_{data} and P_{PFO} that is determined by the two color attributes and by the print media and the print technology.

See also mixing rule.

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.

full-color custom pattern. In GOCA, a custom pattern that has its colors completely assigned during its definition, and can therefore contain any number of colors. Contrast with bilevel custom pattern.

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 • graphics presentation space

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, that interprets commands and instructions, and the drawing processor, that 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.

GZIP. A widely-used, free software compression algorithm.

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’, that 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 Xbc 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, that is also the space character increment

• For PSM fonts: the width of the space character

• For typographic fonts and proportionally spaced fonts: one-third of the vertical font size, that is also the default size of the space character.

The font designer can also define a minimum and 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 in the range X’0001’–X’7EFF’ that is assigned to an IPDS resource by a presentation-services program in the host. This ID uniquely identifies a resource until that resource is deactivated, in which case the HAID can be reused. HAIDs are used in IPDS resource management commands.

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 barcode 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 data, image, audio, and video.

hypertext. Interlinked pieces of information consisting primarily of text.

i. See current inline print coordinate.

i. See initial inline print coordinate.

I. See inline direction.

+I. Positive inline direction.

Ic. See current inline presentation coordinate.

Io. 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 Xp axis of its bounding Xp,Yp coordinate space.

I,B coordinate system. The coordinate system used to present graphic characters. This coordinate system is used to establish the inline direction and baseline direction for the placement of successive graphic characters within a presentation space. See also Xp,Yp coordinate system.

ICC. See International Color Consortium.

ICC profile. A file in the International Color Consortium profile format, containing information about the color reproduction capabilities of a device such as a scanner, a digital camera, a monitor, or a printer. An ICC profile includes three elements: 128-byte file header, tag table, and tagged element data. The intent of this format is to provide a cross-platform device profile format. Such device profiles can be used to translate color data created on one device into another device’s native color space.

ID. Identifier. See also Host-Assigned ID (HAID), correlation ID, font control record, and overlay ID.

IDE. See image data element.

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.
**IM Image.** A migration image object that is resolution dependent, bi-level, 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 **barcode symbology.** In most cases, the range is three to ten cpi. See also **barcode 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 \( I_0 \) and \( B_0. \)

**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 (\( I_0 \)).** The point on the **I axis** where the value of the **inline coordinate** is zero.

**input profile.** An **ICC profile** that is associated with the image and describes the characteristics of the device on which the image was created.

**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 **barcode 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 Color Consortium (ICC).** A group of companies chartered to develop, use, and promote cross-
platform standards so that applications and devices can exchange color data without ambiguity.

**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.

**IPDS dialog.** A series of IPDS commands and IPDS acknowledge replies. An IPDS dialog begins with the first IPDS command that an IPDS device receives and ends either when an IPDS command explicitly ends the dialog or when the carrying-protocol session ends. There can be multiple independent sessions each with an IPDS dialog. See also session.

**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.

**JAN.** See Japanese Article Numbering.

**Japanese Article Numbering (JAN).** The bar code symbology used to code grocery items in Japan.

**jog.** To cause printed sheets 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 Ybc axis of the bar code presentation space. Synonymous with vertical bar code.

**LAN.** See local area network.

**landscape.** A presentation orientation in which the Xm 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 character 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. The line type can either be a standard line type value or a custom line type value. 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 identifier (LID).** An identifier that is mapped by the controlling 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 = 14,400)
- 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 = 1/1440 inch (unit base = 10 inches, units per unit base = 14,400)
- 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**

**M/O.** A table heading for architecture syntax. The entries under this heading indicate whether the construct is mandatory (M) or optional (O).

**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 **character rotation**. 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 **character rotation**. 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**.

**metadata.** Descriptive information that is associated with and augments other data.

**metadata object.** In **AFP**, the resource object that carries **metadata**.

**Metadata Object Content Architecture (MOCA).** A resource object architecture to carry metadata that serves to provide context or additional information about an **AFP** object or other AFP data.

**MFI.** See **mainframe interactive**.

**MICR.** See **magnetic ink character recognition**.

**Microfilm frame.** A rectangular area on the microfilm bounded by imaginary, intersecting grid lines within which a data frame may be recorded. The grid lines are part of gauges used for checking microfilm, but they do not actually appear on the microfilm.

**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.


**MO:DCA.** See **Mixed Object Document Content Architecture**.

**MO:DCA IS/1.** **MO:DCA** Interchange Set 1. A subset of MO:DCA that defines an **interchange** format for presentation documents.
MO:DCA IS/2. **MO:DCA** Interchange Set 2. A retired subset of MO:DCA that defines an *interchange* format for presentation documents.

MO:DCA IS/3. **MO:DCA** Interchange Set 3. A subset of MO:DCA that defines an interchange format for print files that supersedes MO:DCA IS/1.

MO:DCA-L. **MO:DCA** subset that defines the OS/2 Presentation Manager (PM) metafile. This format is also known as a .met file. The definition of this MO:DCA subset is stabilized and is no longer being developed as part of the MO:DCA architecture. It is defined in the document MO:DCA-L: The OS/2 Presentation Manager Metafile (.met) Format, available at www.afpcinc.org.

MO:DCA-P. A subset of the MO:DCA architecture that defines presentation documents. This term is now synonymous with the term MO:DCA.

MOCA. See **Metadata Object Content Architecture**.

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 *modulo*.

modulo. 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*.

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 *nesting resource*.

neutral white. A *color attribute* that gives a device-dependent *default* color, typically white on a screen and black on a printer. Note that neutral white and color of medium are two different colors.

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 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.

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, that can be used to reference the object. Examples of objects are presentation text, font, graphics, and image 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). (1) 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). (2) A variable length (2-bytes long to 129-bytes long) binary ID that uniquely identifies an object. OIDs use the ASN.1 definite-short-form object identifier format defined in the ISO/IEC 8824:1990(E) international standard and described in the MO:DCA Registry Appendix of the Mixed Object Document Content Architecture Reference. An OID consists of a one-byte identifier (X'06'), followed by a one-byte length (between X'00' and X'7F'), followed by 0–127 content bytes.

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-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 Xm axis of the Xm,Ym 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 Xm,Ym 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.

Output profile. An ICC profile that describes the characteristics of the output device for which the image is destined. The profile is used to color match the image to the device’s gamut.
overhead. In a bar code symbology, 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 it receives an IPDS Begin Overlay command.

overpaint. A mixing rule in which the intersection of part of a new presentation space $P_{\text{new}}$ with an existing presentation space $P_{\text{existing}}$ keeps the color attribute of $P_{\text{new}}$. This is also referred to as "opaque" mixing. See also mixing rule. Contrast with blend and 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.

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.

paginated object. A data object that can be rendered on a single page or overlay. An example of a paginated object is a single image in a multi-image TIFF file.

parameter. (1) A variable that is given a constant value for a specified application. (2) A variable used in conjunction with a command to affect its result.

partition. Dividing the medium presentation space into a specified number of equal-sized areas in a manner determined by the current physical media.

partitioning. 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.

pattern. An array of symbols used to fill an area.

pattern attributes. The characteristics that specify the appearance of a pattern.
pattern reference point. In GOCA, a position in the graphics presentation space to be used as the origin of a custom pattern; the pattern is tiled in all directions from this position.

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. (1) See Print Contrast Signal. (2) See Profile Connection Space.

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.

PFB file. A file containing the font information required for presenting the characters of a font. The shape information (glyph procedures) contained in this file is in a binary encoded format defined by Adobe Systems Inc., optimized for small character set fonts having one to two hundred characters (for example, English, Greek, and Cyrillic).

PFO. See preprinted form overlay.

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.

picket fence barcode. A bar code pattern presenting the axis of the symbol in its length dimension parallel to the X 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.

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 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 acknowledgment-required 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.

preprinted form. A form or sheet that is not blank when it is selected as input media for presentation.

preprinted form overlay (PFO). An overlay and associated processing designed to simulate a preprinted form.

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 architectural 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.

printing baseline. A conceptual line with respect to which successive characters are aligned. See also character baseline. Synonymous with baseline and sequential baseline.

print quality. In bar codes, the measure of compliance of a bar code symbol to the requirements of dimensional tolerance, edge roughness, spots, voids, reflectance, PCS, and quiet zones defined within a bar code symbology.

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 $C=0.1$, $M=0.8$, $Y=0.2$, $K=0.1$ 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.

Profile Connection Space (PCS). The reference color space defined by ICC, in which colors are encoded in order to provide an interface for connecting source and destination transforms. The PCS is based on the CIE 1931 standard colorimetric observer.

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.

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.

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.

rasterize. To convert presentation data into raster (bitmap) form for display or printing.

raster pattern. A rectangular array of pixels 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.

Rearranged File. A file containing the mapping of code points to the character index values used in a CID file and to the character names used in one or more PFB files. This is a special case of the CMAP file which permits linking of multiple font files and formats together. The code points conform to a particular character coding system which is used to identify the characters in a document data stream. The mapping information in this file is in an ASCII file format defined by Adobe Systems Inc.

recording algorithm. An algorithm that determines the relationship between the physical location and logical location of image points in image data.

recovery-unit group. In the IPDS architecture, a group of pages identified by the XOH Define Group Boundary command and controlled by the Keep-Group-Together-as-a-Recovery-Unit group operation specified by the XOH Specify Group Operation command. The recovery-unit group also includes all copies specified by the Load Copy Control command.

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.

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 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 device resolution to an image presentation space 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.

RGB. Red, green and blue, the additive primary colors.

RGB color space. The basic additive color model used for color video display, as on a computer monitor.

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 segment 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.

sans serif. A type style characterized by strokes that end with no flaring or crossing of lines at the stroke-ends. Contrast with serif.

SBCS. See single-byte character set.

SBIN. A data type for architecture syntax, that indicates that one or more bytes be interpreted as a signed binary number, with the sign bit in the high-order position of the leftmost byte. Positive numbers are represented in true binary notation with the sign bit set to B'0'. Negative numbers are represented in two-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, 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 a two-way communication with an IPDS device. The session consists of a physical attachment and a communications protocol; the communications protocol carries an IPDS dialog by transparently transmitting IPDS commands and acknowledge replies. See also *IPDS dialog*.

**setup file.** In the *IPDS* architecture, an object container that provides setup information for a printer. Setup files are downloaded in home state and take effect immediately. Setup files are not managed as resources.

**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*.

**shear line direction.** In *GOCA*, the direction specified by the *character shear* and *character angle* attributes.

**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-sheet media, continuous-form media, 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.

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.

Specifications for Web Offset Publications (SWOP). A standard set of specifications for color separations, proofs, and printing to ensure consistency of color printing.

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 controlling environment specifies that processing should continue.

standard line type value. A predefined line type, like solid, invisible, or dash-dot. Contrast with custom line type value.

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, that 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.

subtractive primary colors. Cyan, magenta, and yellow colorants used to subtract a portion of the white light that is illuminating an object. Subtractive colors are reflective on paper and printed media. When used together with various degrees of coverage and variation, they have the ability to
suppression • transform matrix

create billions of other colors. Contrast with additive primary colors.

suppression. A method used to prevent presentation of specified data. Examples of suppression are the processing of text data without placing characters on a physical medium and the electronic equivalent of the "spot carbon," that prevents selected data from being presented on certain copies of a presentation space or a physical medium.

surrogate pair. A sequence of two Unicode code points that allow for the encoding of as many as 1 million additional characters without any use of escape codes.

SWOP. See Specifications for Web Offset Publications.

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, function no longer achievable, 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'06', 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.

system-level font resource. A common-source font from which:

• Document-processing applications can obtain resolution-independent formatting information.

• Device-service applications can obtain device-specific presentation information.

T

temporary baseline. The shifted baseline used for subscript and superscript.

temporary baseline coordinate. The B-value of the I,B coordinate pair of an addressable position on the temporary baseline.

temporary baseline increment. A positive or negative value that is added to the current baseline presentation coordinate to specify the position of a temporary baseline in a presentation space or on a physical medium. Several increments might have been used to place a temporary baseline at the current baseline presentation coordinate.

text. A graphic representation of information. Text can consist of alphanumeric characters and symbols arranged in paragraphs, tables, columns, and other shapes. 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 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, 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 Italic, in any point size.

type family. All characters of a single design, regardless of attributes such as width, weight, posture, and size. Examples are Courier and Gothic.

type structure. Attributes of characters other than type family or typeface. Examples are solid shape, hollow shape, and overstruck.

type style. The form of characters within the same font, for example, Courier or Gothic.

type weight. A parameter indicating the degree of boldness of a typeface. A character’s stroke thickness determines its 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 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 blend and 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 underlining function.

UNDF. A data type for architecture syntax, indicating one or more bytes that are undefined by the architecture.

Unicode. A character encoding standard for information processing that includes all major scripts of the world. Unicode defines a consistent way of encoding multilingual text. Unicode specifies a numeric value, a name, and other attributes, such as directionality, for each of its characters; for example, the name for $ is “dollar sign” and its numeric value is X’0024’. This Unicode value is called a Unicode code point and is represented as U+nnnn. Unicode provides for three encoding forms (UTF-8, UTF-16, and UTF-32), described as follows:

UTF-8 A byte-oriented form that is designed for ease of use in traditional ASCII environments. Each UTF-8 code point contains from one to four bytes. All Unicode code points can be encoded in UTF-8 and all 7-bit ASCII characters can be encoded in one byte.

UTF-16 The default Unicode encoding. A fixed, two-byte Unicode encoding form that can contain surrogates and identifies the byte order of each UTF-16 code point via a Byte Order Mark in the first 2 bytes of the data. Surrogates are pairs of
UTF-16BE • vertical bar code

Unicode code points that allow for the encoding of as many as 1 million additional characters without any use of escape codes.

UTF-16BE
UTF-16 that uses big endian byte order; this is the byte order for all multi-byte data within AFP data streams. The Byte Order Mark is not necessary when the data is externally identified as UTF-16BE (or UTF-16LE).

UTF-16LE
UTF-16 that uses little endian byte order.

UTF-32
A fixed, four-byte Unicode encoding form in which each UTF-32 code point is precisely identical to the Unicode code point.

UTF-32BE
UTF-32 serialized as bytes in most-significant-byte-first order (big endian). UTF-32BE is structurally the same as UCS-4.

UTF-32LE
UTF-32 serialized as bytes in least-significant-byte-first order (little endian).

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 in stores, that can be read and interpreted by a computer.

untoned.
Unmarked portion of a physical medium. Contrast with toned.

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.

US. See Uniform Symbol Specification.

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.

valid 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 character 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 axis.
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.

**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**

**Xbc extent.** The size of a bar code presentation space in the Xbc dimension. See also bar code presentation space.

**Xbc,Ybc 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.

**Xg,Yg 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.

**Xio,Yio coordinate system.** The IO-Image presentation space coordinate system.

**XM.** See Extensible Markup Language.

**XMP.** See Extensible Metadata Platform.

**Xm,Ym coordinate system.** (1) In the IPDS architecture, the medium presentation space coordinate system. (2) In MO:DCA, the medium coordinate system.

**Xo,Yo coordinate system.** The object area coordinate system.

**Xol,Yol coordinate system.** The overlay coordinate system.

**Xp extent.** The size of a presentation space or logical page in the Xp dimension. See also presentation space and logical page.

**Xp,Yp coordinate system.** The coordinate system of a presentation space or a logical page. This coordinate system describes the size, position, and orientation of a
**X<sub>pg</sub>, Y<sub>pg</sub> coordinate system • Y<sub>p</sub> extent**

presentation space or a logical page. Orientation of an X<sub>p</sub>, Y<sub>p</sub> coordinate system is relative to an environment-specified coordinate system. An example of an environment-specified coordinate system is the X<sub>m</sub>, Y<sub>m</sub> coordinate system. The X<sub>p</sub>, Y<sub>p</sub> coordinate system origin is specified by an IPDS Logical Page Position command. See also logical page, medium presentation space, and presentation space.

**X<sub>pg</sub>, Y<sub>pg</sub> 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<sub>pg</sub>, Y<sub>pg</sub> coordinate system is relative to an environment specified coordinate system, for example, an X<sub>m</sub>, Y<sub>m</sub> coordinate system.

**Y**

**Y<sub>bc</sub> extent.** The size of a bar code presentation space in the Y<sub>bc</sub> dimension. See also bar code presentation space.

**Y<sub>p</sub> extent.** The size of a presentation space or logical page in the Y<sub>p</sub> dimension. See also presentation space and logical page.
Index

4-state Customer Barcode ............................................. 75

A
abbreviations ............................................................... xi
acronyms .................................................................... xi
AFP ........................................................................ 142
algorithm, L-unit range conversion.............................. 21
architectures
BCOCA ................................................................. 4, 17
CMOCA ................................................................. 4
FOCA ........................................................................ 4
GOCA ........................................................................ 4
IOCA ........................................................................ 4
IPDS ................................................................. 2, 145
MO:DCA ............................................................... 2, 143
MOCA ...................................................................... 4
PTOCA ...................................................................... 4
Australia Post Bar Code, 25, 27, 33, 35, 37, 40–44, 46, 72, 86, 89, 118, 122, 139
Automatic Identification (AutoID) ................................ 7

B
bar code
AutoID ............................................................... 7
color values ............................................................... 39
definition ............................................................... 7
elements of a system .................................................... 7
in IPDS .............................................................. 145
in MO:DCA documents ............................................. 143
information density .................................................... 13
modifiers ................................................................. 36
presentation space ...................................................... 20
presenting ................................................................. 7
retrieving ................................................................. 7
symbol data definition .............................................. 88
symbol descriptor definition ...................................... 31
symbol generation process ....................................... 13
symbologies ............................................................. 8
symbology specification references ............................ 139
types ...................................................................... 34
bar code commands
in IPDS .............................................................. 145
Bar Code Data Descriptor structured field, MO:DCA ... 144
Bar Code Data structured field, MO:DCA .................. 144
Bar Code Object Content Architecture
compliance ............................................................... 137
definition ............................................................... 17
general concepts ...................................................... 17
generator rules .......................................................... 137
measurements .......................................................... 20
object processor ....................................................... 18
presentation space ...................................................... 20
receiver rules ............................................................ 137
symbol orientation .................................................... 24
symbol placement ..................................................... 23
symbol size ............................................................. 25
Bar Code Symbol Data (BSA) data structure
Code 39 ................................................................. 90
definition ............................................................... 29
flags ...................................................................... 89
format ................................................................. 88

C
check digit calculation methods .................................. 84
check-digit .................................................................. 8, 18
Codabar ............................................................... 26, 34, 36, 43, 45, 55, 85, 118, 121, 139, 141
Code 128 .............................................................. 7, 26, 34, 44–45, 56, 86, 118, 121, 129, 139, 141
Code 39 .............................................................. 7, 13, 26, 34, 36, 43, 45, 47, 84, 90, 118, 120, 139–141
Code 93 .............................................................. 27, 35, 37, 44, 46, 74, 87, 118, 123, 139, 141
color values ............................................................. 39
compliance with BCOCA ........................................... 137
coordinate systems
orthogonal ............................................................... 20
Xbc, Ybc coordinate system ......................................... 20

D
Data Matrix ........................................................... 25–26, 35–36, 38, 42–44, 46, 68, 86, 89, 94, 118, 122, 139
Data Matrix special-function parameters ..................... 94
data structures
BSA ........................................................................ 29
BSD ................................................................. 2, 145
definition ............................................................... 29, 31
diagram, syntax ......................................................... viii
field values calculation ............................................. 21
measurements ........................................................... 21
type data definitions ................................................... viii
data-check exceptions ........................................... 135
decoder ................................................................. 7, 15
Delivery Point bar code ............................................. 64
Dutch KIX ............................................................. 23, 33, 40–41, 46, 65, 86, 92, 118, 122, 140

E
EAN ................................................................. xi, 7, 26–27, 34, 36, 38, 42, 44–45, 52, 62–63, 85, 89, 91, 118, 120, 139–140
element height patterns, BSA .................................... 43
element height, BSD .................................................. 42

Copyright © AFP Consortium 1991, 2015 197
encoding scheme .............................................................. 95, 98, 134
encoding module width technique ........................................... 13
NRZ technique .................................................................. 13
environment, IPDS .............................................................. 145
exception conditions
bar code object processor ...................................................... 18
data-check ........................................................................... 135
EC-0100............................................................................... 131
EC-0200............................................................................... 131
EC-0300............................................................................... 131
EC-0400............................................................................... 38, 131
EC-0500............................................................................... 39, 131
EC-0505............................................................................... 32, 131
EC-0600............................................................................... 40, 131
EC-0605............................................................................... 32, 131
EC-0610............................................................................... 33, 131
EC-0611............................................................................... 33–34, 131
EC-0700............................................................................... 42, 132
EC-0705............................................................................... 32–33, 132
EC-0800............................................................................... 43, 132
EC-0900............................................................................... 43, 132
EC-0A00................................................................................ 92–93, 132
EC-0B00............................................................................... 36, 132
EC-0C00............................................................................... 93, 132
EC-0D00............................................................................... 132
EC-0E00............................................................................... 48, 132
EC-0F00............................................................................... 96–97, 132
EC-0F01................................................................................ 97, 103, 116, 132
EC-0F02................................................................................ 97, 103, 116, 132
EC-0F03................................................................................ 97, 103, 116, 132
EC-0F04................................................................................ 97, 103, 116, 132
EC-0F05................................................................................ 102, 132
EC-0F06................................................................................ 107, 133
EC-0F07................................................................................ 107, 133
EC-0F08................................................................................ 107, 133
EC-0F09................................................................................ 108, 133
EC-0F0A................................................................................ 97–98, 133
EC-0F0B................................................................................ 97–98, 133
EC-0F0C................................................................................ 109, 133
EC-0F0D................................................................................ 109–110, 133
EC-0F0E................................................................................ 114, 134
EC-0F0F................................................................................ 115, 134
EC-0F10................................................................................ 116, 134
EC-0F11................................................................................ 117, 134
EC-0F12................................................................................ 117, 134
EC-1000............................................................................... 89, 134
EC-1100............................................................................... 23, 34, 61, 77, 83, 134, 145
EC-1200............................................................................... 58, 134
EC-1201............................................................................... 99, 134
EC-1202............................................................................... 76, 135
EC-1203............................................................................... 60, 135
EC-1204............................................................................... 83, 135, 149
EC-2100............................................................................... 70, 93, 109, 114, 135
specification-check .............................................................. 131
standard actions .................................................................. 131
Extended Code 39 .............................................................. 47
Extended Code 93 .............................................................. 74
extent .................................................................................. 20, 32–33

G
generator rules .................................................................. 137

H
height multiplier ................................................................. 43
HRI suppression flag .......................................................... 89
HRI, local ID ....................................................................... 38
human-readable interpretation (HRI) ...................................... 19
Human-Readable Interpretation (HRI) guidelines ...................... 26

I
Industrial 2-of-5 ................................................................. 26, 34, 36, 43, 45, 53, 85, 118, 120, 139
information density ............................................................. 13
Intelligent Mail Barcode ..................................................... 23, 25, 27, 33, 35, 37, 40–44, 46, 75, 87, 92, 118, 123, 140
Intelligent Mail Container Barcode ..................................... 26, 36, 45, 56, 89, 121
Intelligent Mail Tray Barcode .............................................. 56
Intelligent Printer Data Stream additional related commands .............................................................. 147
bar code command set .......................................................... 145
bar code object area ............................................................. 18
BCOCA Exception Conditions and IPDS Exception IDs ........... 148
environment ....................................................................... 145
intercharacter gap .............................................................. 18
Interleaved 2-of-5 .............................................................. 7, 13, 26, 34, 36, 43, 45, 53, 85, 118, 120, 139–141
IPDS
Intelligent Printer Data Stream ........................................... xi, 26, 34, 36, 43, 45, 53, 85, 118, 120, 140

J
Japan Postal Bar Code ......................................................... 25–26, 34, 36, 38, 42–44, 46, 66, 86, 89, 118, 122, 140

L
ladder orientation ............................................................... 24
LID ...................................................................................... 38
linear measurements ........................................................... 20
linear symbologies ............................................................ 8
local ID ............................................................................. 38
logical units (L-units)
BSD, x direction ................................................................. 32
BSD, y direction ................................................................. 32
definition ........................................................................... 20
field values calculation ...................................................... 21
fields ................................................................................. 21
presentation space ............................................................ 20

M
Matrix 2-of-5 ............................................................... 26, 34, 36, 43, 45, 53, 85, 118, 120, 139
MaxiCode ......................................................... 23, 25–26, 33, 35–36, 38, 40–44, 46, 69, 86, 89, 92, 100, 118, 122, 139
MaxiCode special-function parameters ................................ 100
measurement base ........................................................... 20, 32
measurements .................................................................. 20
message ........................................................................... 8
metadata .......................................................................... 4
<table>
<thead>
<tr>
<th>UPC .................................................................</th>
<th>xi, 7, 19, 26–27, 34, 36, 38, 42, 44–45, 49–51, 85, 89, 91, 118, 120, 139–141</th>
</tr>
</thead>
<tbody>
<tr>
<td>USPS Four-State .....................................................</td>
<td>75</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>W</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>WE:NE ..........................................................</td>
<td>13, 43</td>
</tr>
<tr>
<td>wide-to-narrow ratio ........................................</td>
<td>13, 43</td>
</tr>
<tr>
<td>Write Bar Code command ......................................</td>
<td>146</td>
</tr>
<tr>
<td>Write Bar Code Control command ............................</td>
<td>145</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_{bc}, Y_{bc}$ coordinate system ........................</td>
<td>20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Z</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>zipper ..................................................</td>
<td>18, 23, 100, 103–104, 134, 146</td>
</tr>
</tbody>
</table>