Data Stream and Object Architectures

Image Object Content Architecture Reference

Release 6.0

S550-1142-00

RICOH | IBM

InfoPrint Solutions Company

Data Stream and Object Architectures

Image Object Content Architecture Reference

Release 6.0

S550-1142-00

RICOH | IBM

InfoPrint Solutions Company

Note:
Before using this information and the product it supports, read the information in "Notices" on page 175.

First Edition (August 2008)

This edition replaces SC31-6805-05.

This publication is available from the InfoPrint Solutions Company or the AFP Consortium Web site at

http://www.afpcolor.org

Internet
Visit our home page: `http://www.infoprint.com`

You can send comments by e-mail to **`printpub@us.ibm.com`** or by mail to:

InfoPrint Solutions Company
6300 Diagonal Hwy 002J
Boulder, CO 80301-9270
U.S.A.

About this book

This book describes the functions and services associated with Image Object Content Architecture (IOCA). It 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 develop or adapt a product or program to interoperate with other presentation products in an Advanced Function Presentation™ environment.

How to use this book

This book contains the following sections:

- Chapter 1, "A Presentation Architecture perspective," provides a brief overview of Presentation Architecture.
- Chapter 2, "Introduction to IOCA," discusses the background of image processing and introduces IOCA.
- Chapter 3, "IOCA overview," discusses concepts involved in image processing.
- Chapter 4, "Formats and codes," shows formats used by IOCA, and code points assigned to and reserved for IOCA.
- Chapter 5, "IOCA image segment," describes the components of the IOCA entity.
- Chapter 6, "Exception conditions and actions," lists exceptions to the IOCA definitions, and standard actions to take when exceptions occur.
- Chapter 7, "Compliance," describes the function sets that IOCA defines.
- Appendix A, "Compression and recording algorithms," discusses compression and recording algorithms that IOCA supports.
- Appendix B, "Bilevel, grayscale, and color images," summarizes how to specify these different types of images.
- Appendix C, "IOCA Tile Resource," describes the structure and use of tile resources.
- Appendix D, "MO:DCA environment," describes how the IOCA image segments are carried in the MO:DCA data stream controlling environment.
- Appendix E, "IPDS environment," describes how the IOCA image segments are carried in the IPDS™ architecture controlling environment.
- Appendix F, "Notes for IOCA generators," discusses issues that should considered when generating efficient IOCA for high speed printing.
- Glossary defines terms used in this book.

How to read the syntax diagrams

Throughout this book, syntax for IOCA is shown in tables, laid out as follows:

Offset	Type	Name	Range	Meaning	M/O
Byte offset	Data type	Name, if applicable	Range of valid values, if applicable	Meaning or purpose of the parameter	M or O
Bit offset					

The "M/O" column indicates whether the parameter is mandatory or optional.

The syntax includes the following basic data types:

BITS	Bit string
CHAR	Character string
CODE	Architected constant
UBIN	Unsigned binary

The following is an example of IOCA syntax.

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'9B'	IDE Structure parameter	M
1	UBIN	LENGTH	X'06' – X'09'	Length of the parameters to follow	M
2	BITS	FLAGS			M
Bit 0		ASFLAG	B'0' – B'1'	Additive or subtractive: **B'0'** Additive **B'1'** Subtractive	
Bit 1		GRAYCODE	B'0' – B'1'	Gray coding: **B'0'** Off **B'1'** On	
Bits 2–7			B'000000'	Reserved; should be zero	
3	CODE	FORMAT	X'01' – X'02' X'04' – X'12'	Color model: **X'01'** RGB **X'02'** YCrCb **X'04'** CMYK **X'12'** YCbCr All other values are reserved.	M
4–6			X'000000'	Reserved; should be zero	M
7	UBIN	SIZE1	X'00' – X'FF'	Number of bits/IDE for component 1	M
8	UBIN	SIZE2	X'00' – X'FF'	Number of bits/IDE for component 2	O
9	UBIN	SIZE3	X'00' – X'FF'	Number of bits/IDE for component 3	O
10	UBIN	SIZE4	X'00' – X'FF'	Number of bits/IDE for component 4	O

Notation conventions

Throughout this document, the following notation conventions apply:

- Bytes are numbered from left to right beginning with byte 0, which is considered the high order byte position. For example, a three-byte field consists of byte 0, byte 1, and byte 2.
- Each byte is composed of eight bits.
- Bits in a single byte are numbered from left to right beginning with bit 0, the most significant bit, and continuing through bit 7, the least significant bit.
- When bits from multiple consecutive bytes are considered together, the first byte, byte 0, contains bits 0 to 7, and byte *n* contains bits $n \times 8$ to $n \times 8+7$.
- A negative number is expressed by the two's-complement form of its positive number. The two's complement of a number is obtained by first inverting every bit of the number and then adding one to the inverted number.

In the syntax summary diagrams, the conventions in the parameter groupings are:

- The identifier is shown for all the parameters. If the identifier is missing, the item is not a parameter, but a grouping of parameters, for example, a tile.
- The following symbols have special meanings:

[ ]	Brackets indicate optional parameters. When a parameter is shown without brackets, it *must* appear if the corresponding grouping is present. For example, if a tile is being specified, Tile Position must appear.
+	Plus signs indicate that a group of successive parameters may appear in any order relative to each other.
(S)	The enclosed (S) indicates that the parameter may be repeated. When it is present on a required parameter, at least one instance of the parameter is required, but multiple instances of it may occur.

Changes to the architecture

Summary of changes

The following changes have been made to this edition:

- Function Set 00 has been retired, together with the old tiling scheme.
- A new tiling scheme was formulated.
- Support for high speed color printing was added. This includes CMYK color space, distinction between continuous tone and linework data handling, and the LZW compression algorithm.
- Support for transparency masks has been added.
- Multiple Image Contents are now allowed within a single Image Segment.
- New function sets 40, 42 and 45 were added. These function sets are *tiled function sets*. Function Set 40 is a bilevel tiled function set. Function Set 42 supports color printing using one bit per color, while Function Set 45 supports color printing with eight bits per color.
- Appendix C, "IOCA Tile Resource," on page 143 has been added.
- Appendix F, "Notes for IOCA generators," on page 171 has been added.
- The External Algorithm Specification Parameter was made optional for the JPEG compression algorithm, instead of being mandatory.
- Numerous editing and formatting changes were made, including a number of new figures and architecture summary listings.

| Changes are marked with the change bar.

Contents

Figures

Tables

Chapter 1. A Presentation Architecture perspective

This chapter provides a brief overview of the Advanced Function Presentation (AFP™) Architecture.

The presentation environment

Figure 1 shows today's presentation environment.

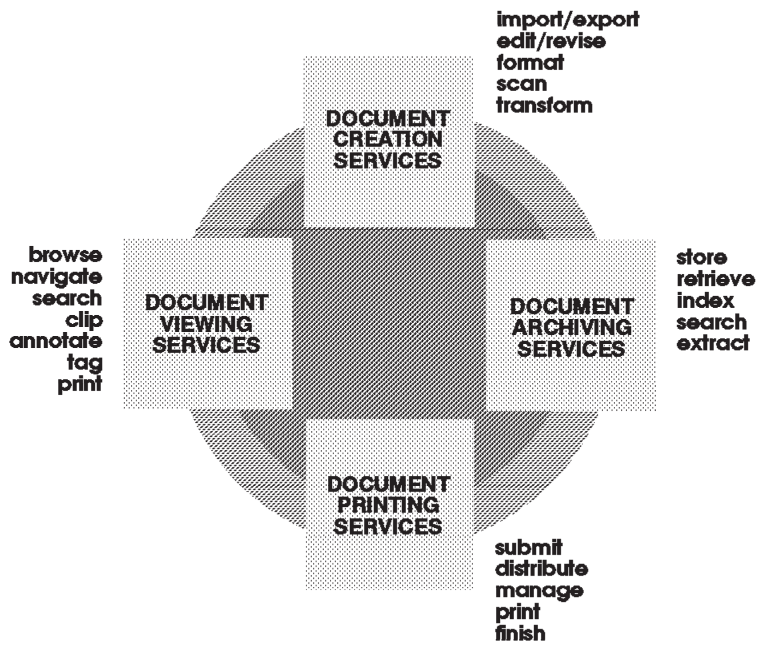


Figure 1. Presentation environment

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 facsimile standards for compressed image data established by the International Telecommunications Union–Telecommunication Standardization Sector (ITU–TSS), formerly known as the Comité Consultatif International Télégraphique et Téléphonique (CCITT).

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 can contain combinations of text, image, graphics, and bar code objects in device-independent and resolution-independent formats. Documents can contain fonts, overlays, and other resource objects required at presentation time to present the data properly. Finally, documents can contain resource objects, such as a document index and tagging elements supporting the search and navigation of document data, for a variety of application purposes.

In AFP, 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*

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

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

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

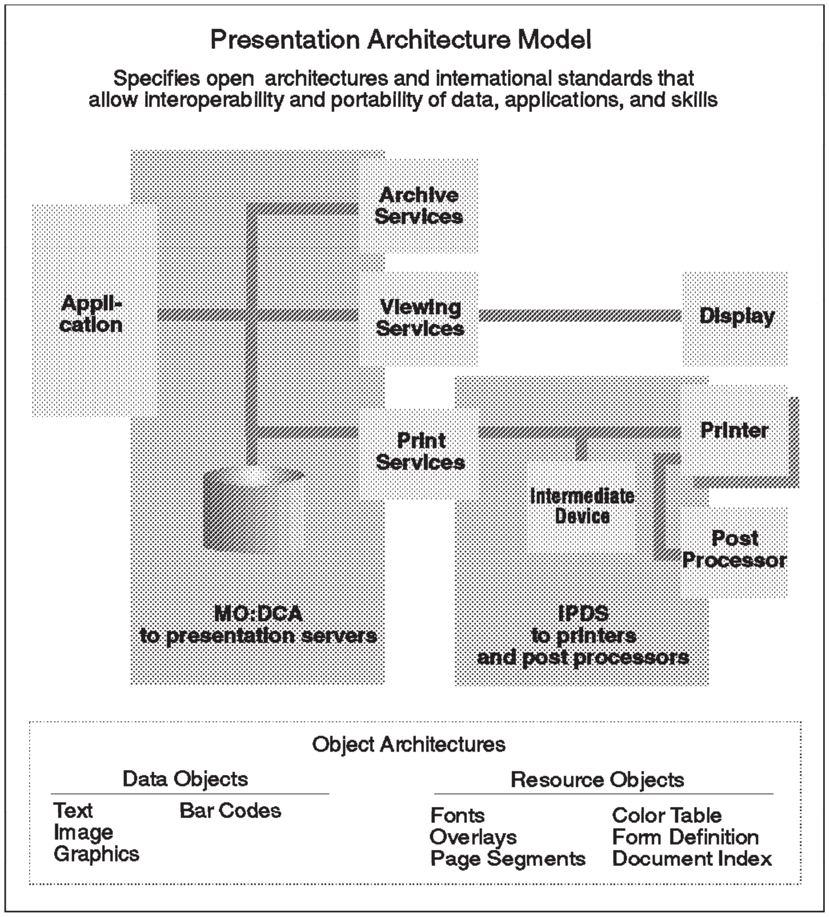


Figure 2. Presentation model

Objects

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

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

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

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

The object content architectures are:

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

The MO:DCA and IPDS architectures also support data objects that are not defined by AFP 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, MO:DCA 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.

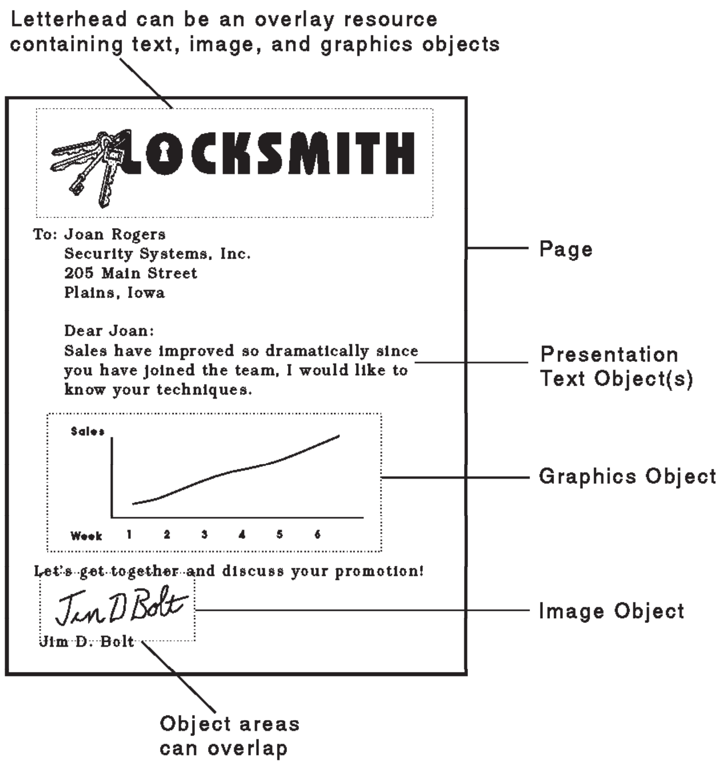

Letterhead can be an overlay resource containing text, image, and graphics objects
LOCKSMITH
To: Joan Rogers
Security Systems, Inc.
205 Main Street
Plains, Iowa
Dear Joan:
Sales have improved so dramatically since you have joined the team, I would like to know your techniques.
Sales
Week 1 2 3 4 5 6
Let's get together and discuss your promotion!
Jim D. Bolt
Page
Presentation Text Object(s)
Graphics Object
Image Object
Object areas can overlap

Figure 3. Presentation page

Chapter 2. Introduction to IOCA

This chapter outlines:
- The rationale for IOCA
- The scope of IOCA

Background

An *image*, in computer terminology, is an electronic representation of a picture as an array of raster data. Image data can be generated by a computer program, or formed by electronically scanning such items as illustrations, drawings, photographs, and signatures.

The image-processing field is expanding dramatically due to advances in hardware technology. For example:
- Less expensive computer storage and memory are making the handling of larger volumes of image data increasingly more feasible; image databases are now in widespread use.
- Faster processors and techniques such as bit slicing and hardware buffering are improving the efficiency and flexibility of online image processing.
- Higher-resolution image devices are improving the usability of images and image applications. Images can now be printed and displayed in greater detail than ever before.

More and more image applications—most of which involve generating, processing, presenting, and storing images—are emerging to meet the specific needs of various industries. Insurance applications often require high-volume input and single-image manipulation. Banking applications require a verification process for handwritten check endorsements and signatures, with the ability to analyze a specific part of each image. Engineering applications may focus on design analysis systems that deal with drawings. Publishing applications may involve document creation, complex editors with image editing capabilities, and document distribution. The list of potential areas for image applications is very long and continues to grow: medicine, geology, agriculture, manufacturing, and government, to name a few.

To support the diverse image application areas, images are encoded in a number of different formats. As the technology progresses, old formats are extended and refined and new formats are being formulated.

The Image Object Content Architecture (IOCA) has been formulated to provide a format suited for high speed printing. IOCA contains enough flexibility that a wide variety of images can be printed, but formats images in such a way that they can be printed efficiently and with minimal processing.

What is IOCA?

IOCA is an architecture that provides a consistent way to represent images, including conventions and directions for processing and interchanging image information. In other words, this architecture:

- Can be used for scanning, displaying, printing, archiving, and other I/O operations.
- Has an image description which is flexible enough to allow it to exist intact in interactive, printer, and interchange environments that are defined by the following data stream architectures:
  - Intelligent Printer Data Stream (IPDS) for printers
  - Mixed Object Document Content Architecture (MO:DCA)
- Allows the image to be fully described in device- and process-independent terms. Each image object is independent of other data objects and the environment in which it exists.
- Describes images using self-defining fields; that is, each field contains a description of itself along with its contents.

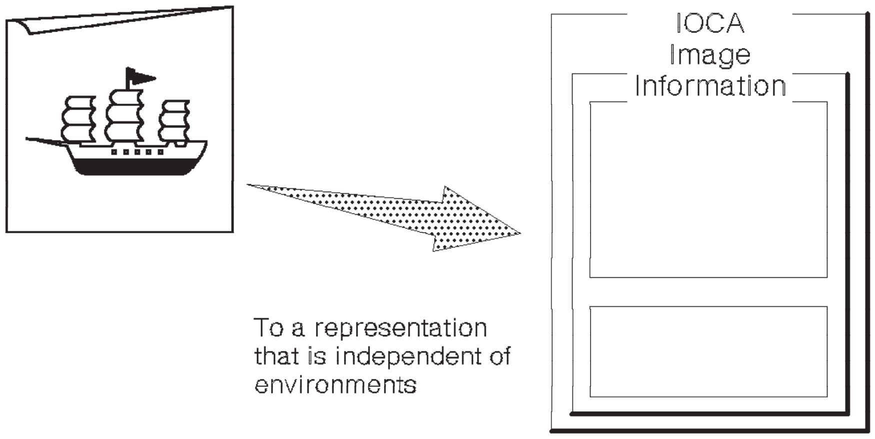


Figure 4. Images and IOCA

IOCA in image processing

Figure 5 summarizes the steps typically involved in image processing, and indicates which stages are device-dependent. IOCA is involved only in Step 3 on page 9, device-independent information processing. The term *IOCA process model* is used hereafter when referring to this step. The other steps are device-dependent, and the interface to them is provided by the *controlling environments*.

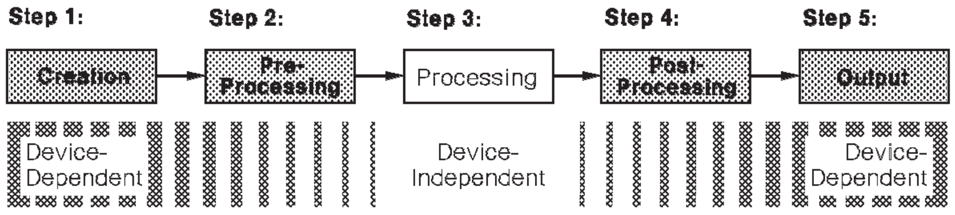


Figure 5. Steps in image processing

1. *Creation*. An image is created by a program or an input device such as a scanner. The creation step is supported by many types of devices and technologies. The resulting image contains device-dependent information.
2. *Preprocessing*. Preprocessing is the gateway from the input devices. In this step, the device-dependent information is removed from the image. For example, if the image was created by scanning a document, the end-of-scan-line code is removed. After this step, the image, along with its characteristics of resolution and size, is ready to be processed.
3. *Processing*. The image is now processed into an interchangeable form with all device-dependent characteristics removed. In this form, it can be passed to another system or environment and interpreted consistently.
4. *Postprocessing*. Postprocessing is the gateway to applications that support output devices. The required device-control information is inserted. This step might be different for each type of device.
5. *Output*. This step presents the image to the user. It is controlled locally by the output device, such as a display or a printer.

The IOCA process model

IOCA uses the *image segment* as its base unit for representing an image. An image segment consists of image data and the parameters needed to describe that image's characteristics in a universally recognizable way.

The IOCA process model communicates with the controlling environment, sending and receiving image segments to and from them. It also takes action if irregularities are found in the IOCA image segments.

Figure 6 shows the relationship between the IOCA process model and the controlling environments that scan, display, and print IOCA image segments.

IOCA process model

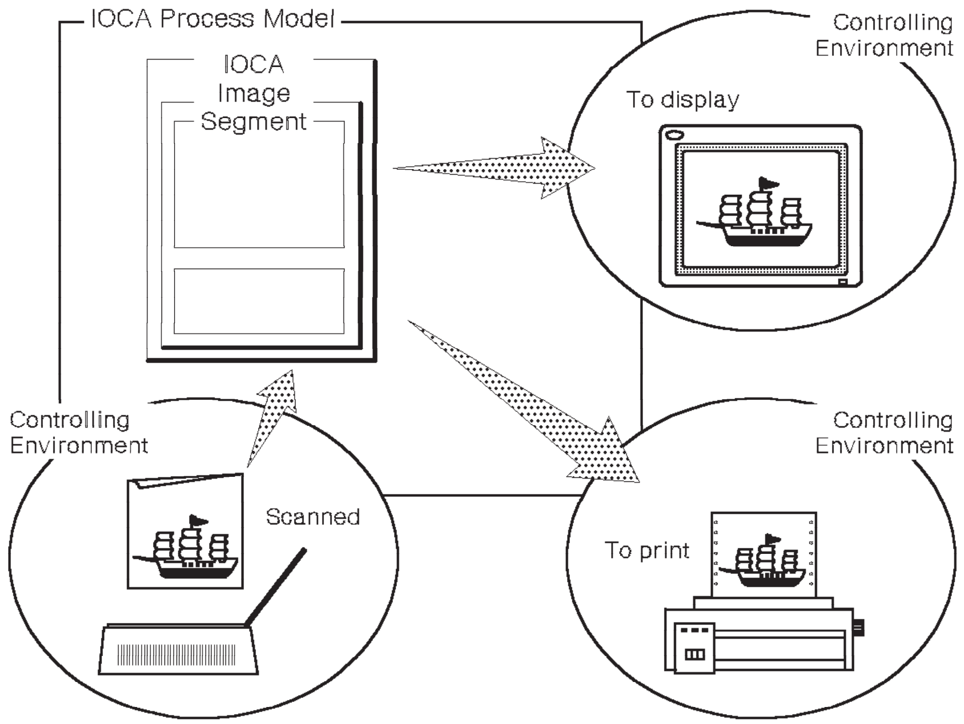


Figure 6. IOCA process model and the controlling environments

Mixed Object Document Content Architecture (MO:DCA) and Intelligent Printer Data Stream (IPDS) are examples of controlling environments.

Chapter 3. IOCA overview

This chapter outlines:
- IOCA representation of image attributes
- Compression
- The image coordinate system
- The image presentation space
- Image tiling
- Function sets

IOCA representation of images

IOCA provides a way to represent images in a device-independent format, which allows them to be interchangeable across environments. IOCA uses a consistent set of constructs, called *self-defining fields*, to describe the characteristics of the image data. A self-defining field is a field that contains one or two bytes identifying the content of that field.

An image consists of *image points*. Each image point is represented by one or more bits of information, called *image data elements* (IDEs). IDEs are grouped together into *image data*. Image data is known as non-coded information (NCI) because no codes are embedded in it. This characteristic makes image data different from either text or graphic data.

Certain properties characterize the image, and must be processed in order to interpret the data properly, such as:
- Size (how large)
- Resolution (how sharp)
- Color (whether it is black-and-white, grayscale, or color)
- Recording and compression algorithms (how image data is encoded)
- Image data layout

Image data parameters encapsulate these properties and separate them from the image data. The image data and image data parameters are collectively referred to as the *image content*.

The image contents are independent of the controlling environment in which they exist. In every controlling environment, an image can be represented by its image contents alone.

When an image is carried in data streams, all of its image components are contained in *image segments*.

The image segment, a set of self-defining fields, is passed to and from controlling environment, which determine how it is handled. That is, the image segment can be presented as a displayed or a printed image in an environment, or can be merged with text and graphics objects into a compound document.

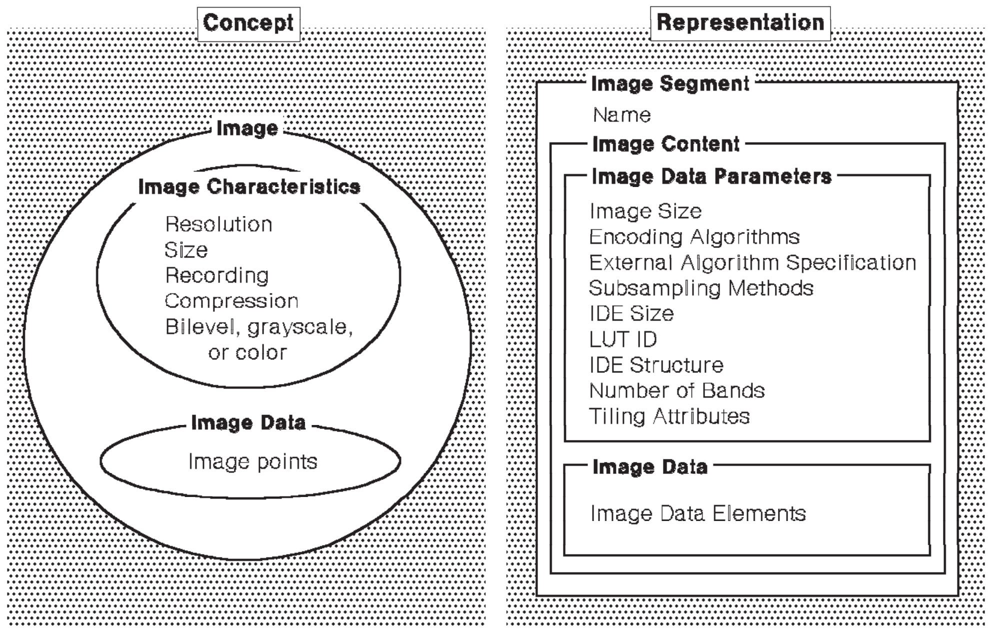


Figure 7. Image concept and IOCA representation

Image points

When digitized for processing, images are expressed by a two-dimensional array of pixels, called *image points*.

Each image point has information called the *image data element* (IDE). The IDE has one or more bits that refer to a *look-up table* (LUT). With this table, each IDE value is interpreted to determine its property, such as black, white, grayscale, or color.

Consider a color image which is represented by three bits per IDE. Figure 8 on page 13 shows how an intensified image point, say IDE with a binary value of B'100', is interpreted.

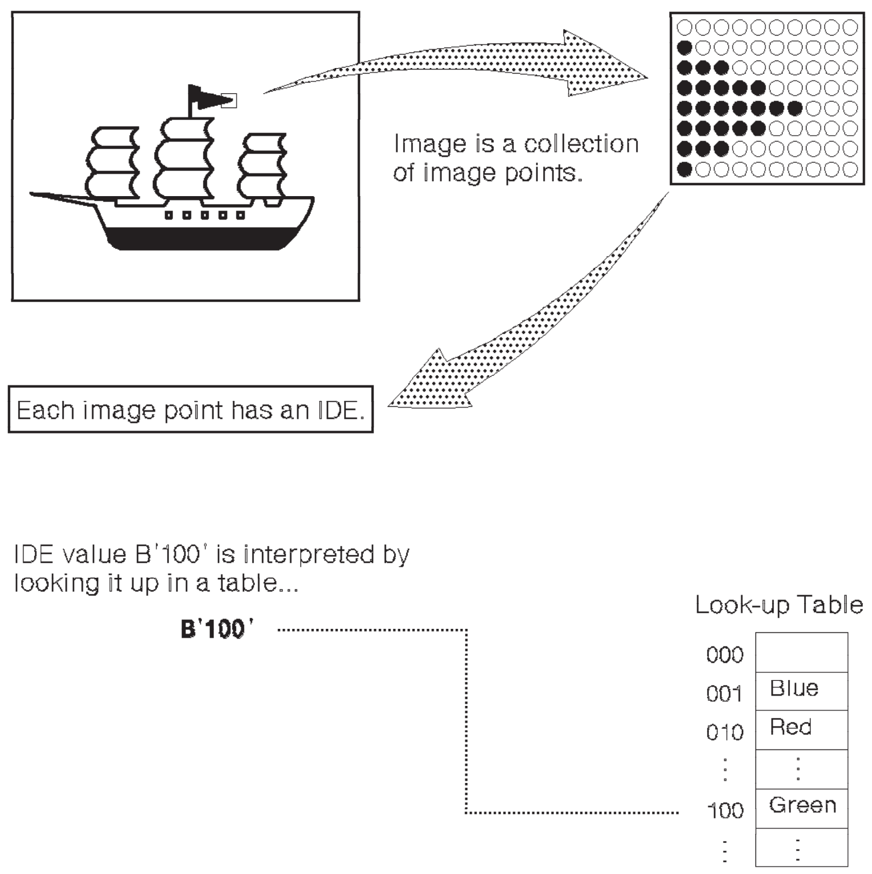


Image is a collection of image points.
Each image point has an IDE.
IDE value B'100' is interpreted by looking it up in a table...
B'100'
Look-up Table
000
001 Blue
010 Red
100 Green

Figure 8. Image point, IDE, and LUT-ID

The image foreground and background are defined as follows:

- For bilevel images with no explicit color table defined, the image foreground consists of all those image points whose IDE values are B'1'. The rest of the image points along with the unoccupied areas of the image presentation space (IPS) are considered to be the image background.
- For any other images (including bilevel images with explicit defined color table, that is, non-zero LUT-ID), the entire image is considered to be foreground. The unoccupied areas in the image presentation space are the image background.

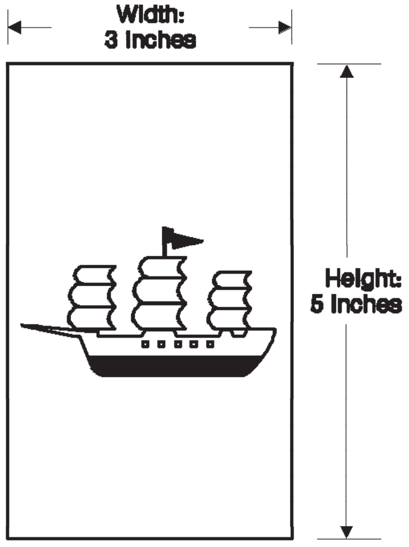
Size and resolution

In addition to color, images are characterized by their size and resolution.

- The size of an image is expressed in terms of the number of image points in the horizontal and vertical directions.
- The resolution of an image determines its sharpness. It is expressed in terms of the number of image points in the measurement base, in the horizontal and vertical directions. The measurement base, indicated by *unit base*, can be 10 inches or 10 centimeters.

Size and resolution

Figure 9 shows how an image's size and resolution are calculated:



If the image is divided into 600 image points horizontally and 1500 image points vertically, the image is represented as:

- Sizes:

600	Horizontal
1500	Vertical

- Resolutions:

  200 points/inch
  Horizontal

  300 points/inch
  Vertical

Figure 9. Image resolution

Compression

Consider an image that has the dimensions of letter-size paper. If it is represented in black and white (bilevel, represented by one bit per IDE) at 600 dpi, its image data would be about 3,366,000 bits long. Such large data volumes are expensive to process, store, and transmit.

The size of an image's data can be reduced by one of many compression techniques. In order to reconstruct a compressed image, an application or device must know which compression technique was used to compress the data. IOCA provides two self-defining fields to describe the compression algorithm.

In the image data, it is not unusual to find lengthy strings of IDEs that all have the same value. Compression algorithms use codes to represent these strings in the image data.

Figure 10 shows a compression example that takes advantage of IDE repetitions in the image data. The compression algorithm represents a group of similar IDEs by the length of that group.

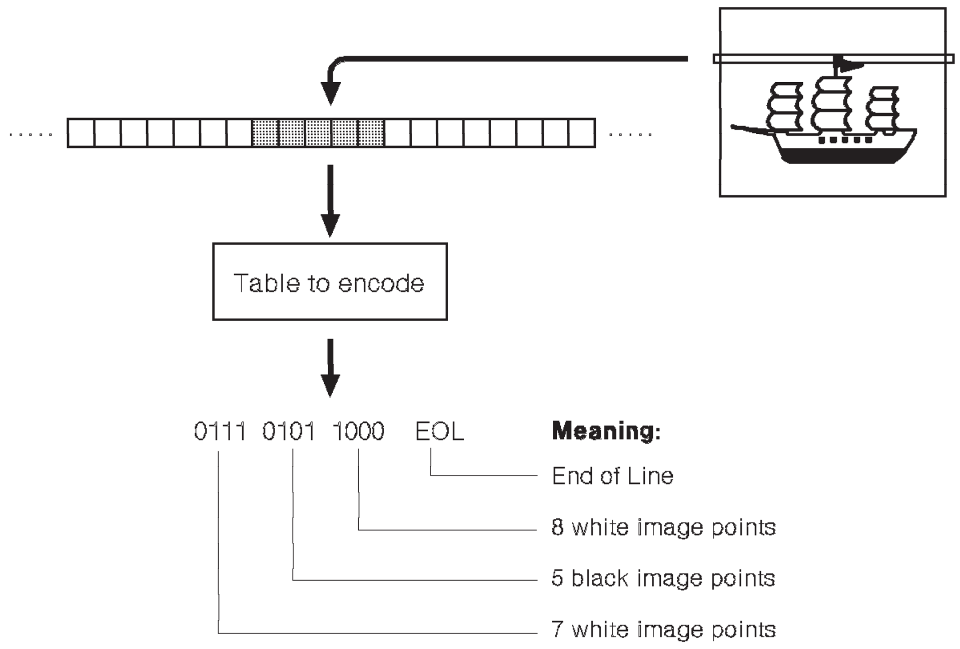


Figure 10. Image compression

The effectiveness of compression algorithms differs depending on the content of the image. For example, text, which has a great deal of white space, compresses well; maps, complicated drawings with many transitions of black and white, as well as photographic images, do not compress well.

Image coordinate system

Each image content, which consists of image data and image characteristics information, has a coordinate system, called the *image coordinate system*. This is an X-Y Cartesian system that uses only the fourth quadrant and positive values for the Y-axis. In other words, the origin is top left. Units along the X and Y axes correspond directly to image points that are represented by IDEs in the image content.

Image coordinate system

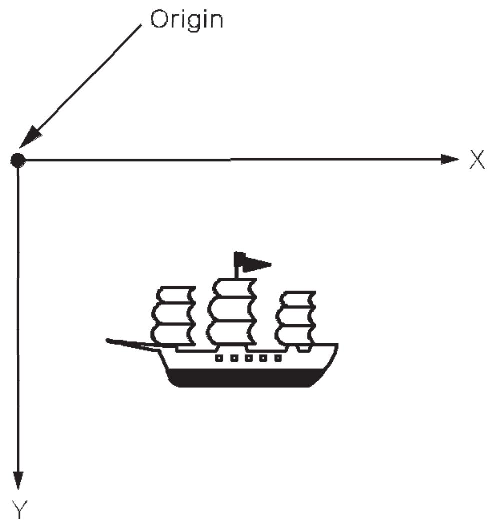


Image points in the horizontal direction are mapped in the X direction of the image coordinate system.

Image points in the vertical direction are mapped in the Y direction of the image coordinate system.

Figure 11. Image coordinate system

Image presentation space

Before an image content can be displayed or printed, it is placed in a conceptual space, called an *image presentation space* (IPS). The physical characteristics of the IPS are defined and provided by the controlling environment. The IPS is two-dimensional, and has an image coordinate system. It acts as a bridge between the IOCA process model and the controlling environment.

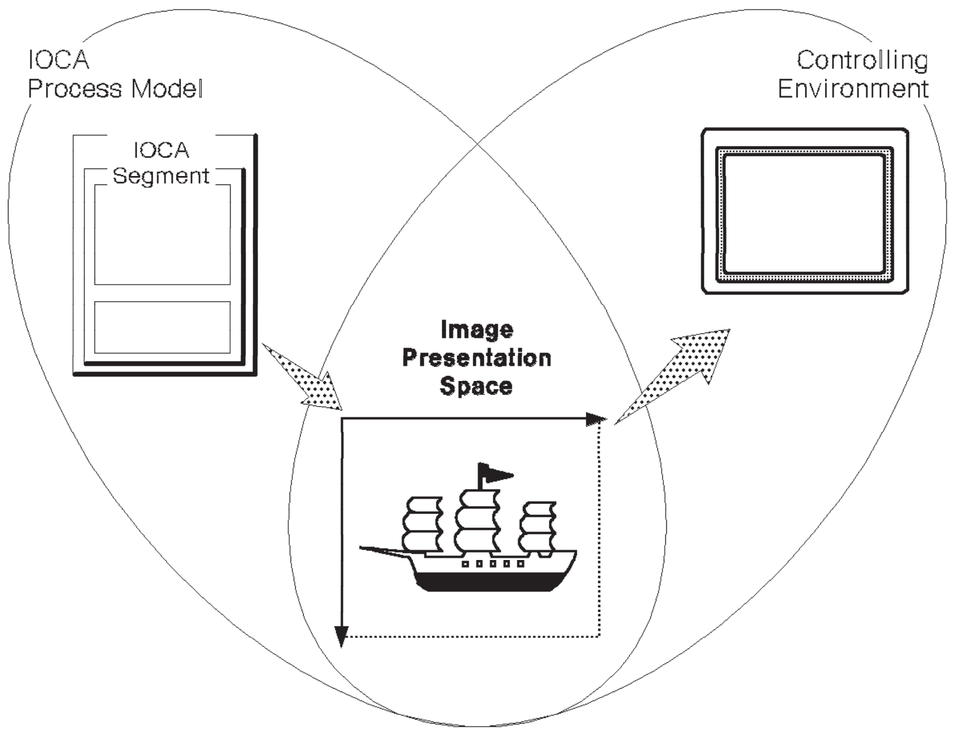


Figure 12. Image presentation space

Image tiling

For large images, such as engineering drawings, it is often advantageous to partition the image into smaller non-overlapping rectangular pieces called *tiles*.

Each tile can be thought of as an individual image. The tiles may differ in the color space, encoding and compression algorithms, but must have resolution that evenly divides the underlying image presentation space resolution. The tiles need not cover the whole image presentation space.

IOCA provides a series of self-defining fields to encode tiling information.

Figure 13 illustrates an image composed of three tiles, each with a different data type.

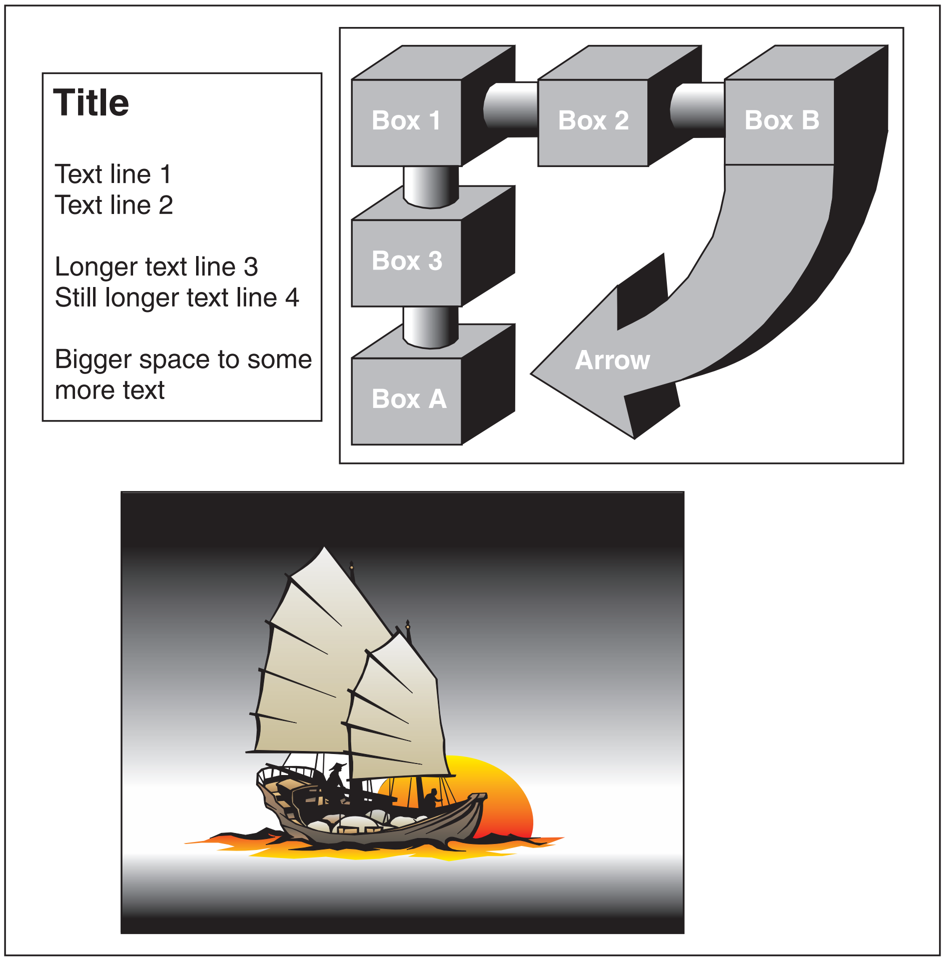


Figure 13. Tiles of an image

Function sets

For some applications, it is not necessary or feasible to implement all the features in the architecture, or support the entire range of values and parameters in a self-defining field.

Chapter 7, “Compliance,” on page 91 defines several subsets of the architecture (called *function sets*) that satisfy some particular common needs. It is the responsibility of the application to determine which function sets it must provide to generate and receive IOCA image objects.

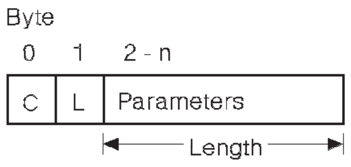
Chapter 4. Formats and codes

This chapter describes the formats of the IOCA self-defining fields.
- The formats of the IOCA self-defining fields
- The code points used by IOCA

Formats

An IOCA image segment is a set of self-defining fields. Each self-defining field is in either long format or extended format. Both formats start with a code for the self-defining field, and the length of the parameters that follow.

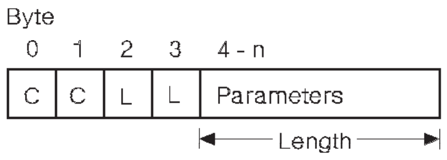
Long format



where:

C	is a one-byte code for the self-defining field.
L	is the length of the following parameters, excluding L itself.

Extended format



where:

CC is a two-byte code for the self-defining field. The first byte is always X'FE'.

This format is used by all of the following:
- Image data (X'FE92')
- Band image data (X'FE9C')
- Include Tile parameter (X'FEB8') |
- Tile TOC parameter (X'FEBB') |
- Image Subsampling parameter (X'FECE').

Other values for the second byte of CC are reserved.

LL is the length of the parameters, excluding LL itself.

Code points

Table 1 lists the codes used by IOCA, the names of the associated elements, and the formats used.

Table 1. IOCA code points

Code	Name	Format
X'70'	Begin Segment	Long format
X'71'	End Segment	Long format
X'8C'	Begin Tile	Long format
X'8D'	End Tile	Long format
X'8E'	Begin Transparency Mask	Long format
X'8F'	End Transparency Mask	Long format
X'91'	Begin Image Content	Long format
X'93'	End Image Content	Long format
X'94'	Image Size parameter	Long format
X'95'	Image Encoding parameter	Long format
X'96'	IDE Size parameter	Long format
X'97'	Image LUT-ID parameter	Long format
X'98'	Band Image parameter	Long format
X'9B'	IDE Structure parameter	Long format
X'9F'	External Algorithm Specification parameter	Long format
X'B5'	Tile Position	Long format
X'B6'	Tile Size	Long format
X'B7'	Tile Set Color	Long format
X'F6'	Set Bilevel Image Color	Long format
X'F7'	IOCA Function Set identification	Long format
X'FE92'	Image data	Extended format
X'FE9C'	Band image data	Extended format
X'FEB8'	Include Tile	Extended format
X'FEBB'	Tile TOC	Extended format
X'FECE'	Image Subsampling parameter	Extended format

Chapter 5. IOCA image segment

This chapter:

- Briefly describes the IOCA image segment
- States the purpose of each IOCA self-defining field in the image segment
- Provides the syntax and semantics of each self-defining field, its parameter set, and its exception conditions

For an explanation of the layout of the syntax diagrams in this chapter, see “How to read the syntax diagrams” on page iv. For an explanation of the notation conventions, see “Notation conventions” on page iv.

Image segment

An image segment is represented by a set of self-defining fields, fields that describe their own contents. It starts with a Begin Segment, and ends with an End Segment.

Between the Begin Segment and End Segment is the image information to be processed, called the image content.

The image content can be either untiled or tiled.

Untiled image content consists of:

- Image data parameters, which describe the characteristics of the image data
- An optional transparency mask
- Zero or more image data elements: Image Data and Band Image Data.

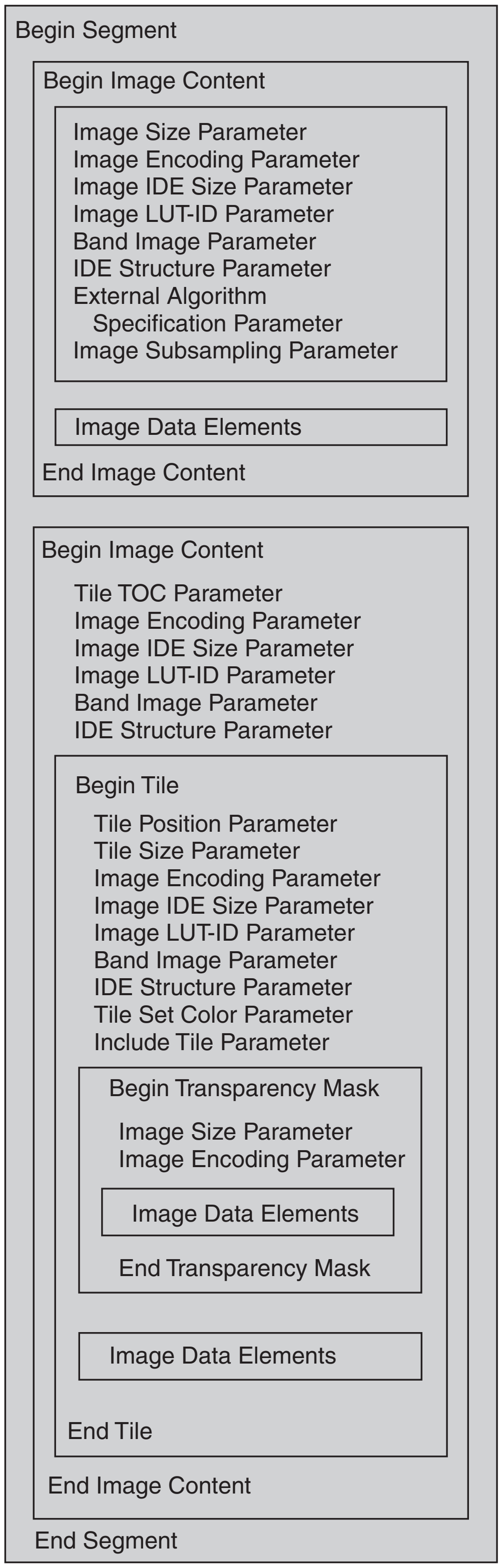
Tiled image content consists of:

- Image data parameters, which describe the characteristics of the image content.
- Zero or more tiles.

Each tile consists of:

- Image data parameters, which describe the characteristics of the image data.
- An optional transparency mask.
- Zero or more image data elements: image data and band image data.

Multiple image contents can exist within a single IOCA image segment. All image contents share the same image presentation space and are presented in the order they appear.



Begin Segment

The Begin Segment parameter defines the beginning of the image segment.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'70'	Begin Segment	M
1	UBIN	LENGTH	X'00' – X'04'	Length of the parameters to follow	M
2	UBIN	NAME	X'00000000' – X'FFFFFFFF'	Name of the image segment	O

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-0005 **Invalid Length**

Condition: The LENGTH value is not in the valid, function-set specified range. EC-0005 is optional: IOCA receivers can generate EC-0003 instead of EC-0005.

EC-700F **Invalid sequence**

Condition: A Begin Segment is missing, or it appeared out of sequence or more than once. IOCA receivers can generate an out-of-sequence exception condition, EC-*xx*0F, instead of EC-700F, where *xx* is the one-byte ID code of the IOCA self-defining field encountered in place of the Begin Segment self-defining field.

End Segment

The End Segment parameter defines the end of the image segment.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'71'	End Segment	M
1	UBIN	LENGTH	X'00'	Length of the parameters to follow	M

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-710F **Invalid sequence**

Condition: An End Segment is missing, or it appeared out of sequence.

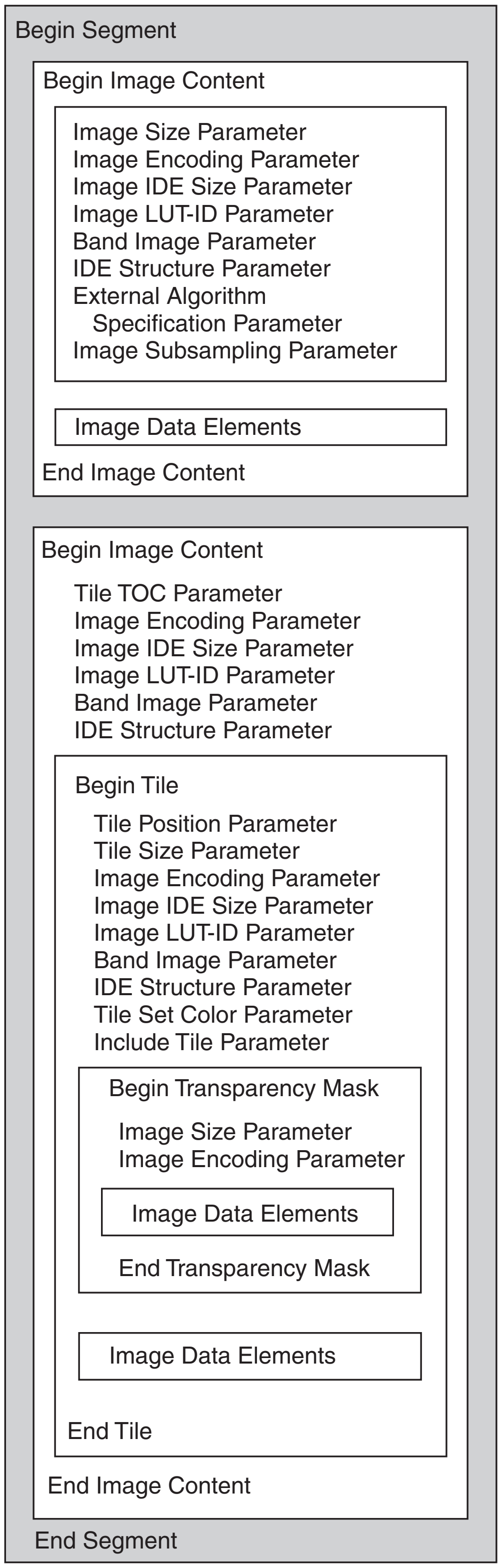
Image content

An image content begins with a Begin Image Content and ends with an End Image Content.

The image content can be either untiled or tiled.

If the image content is untiled, it contains a number of image data parameters, followed by the image data. The image data is contained in one or more self-defining fields. The same image data parameter cannot appear more than once within a single image content.

If the image content is tiled, it optionally starts with a number of parameters that set the default values, followed by zero or more tiles.



Begin Image Content

The Begin Image Content parameter defines the beginning of the image content.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'91'	Begin Image Content	M
1	UBIN	LENGTH	X'01'	Length of the parameters to follow	M
2	CODE	OBJTYPE	X'FF'	Object type: **X'FF'** IOCA image object All other values are reserved.	M

Notes:

1. IOCA allows multiple image contents in a single image segment, but the receivers are not required to support more than one image content in each image segment. If a receiver that does not support multiple image contents in a single image segment receives a second Begin Image Content parameter in an image segment, exception EC-910F exists.
2. All receivers that support multiple image contents must support at least 128 image contents per image segment.
3. Architecture does not restrict the number of image contents contained within a single image segment. If an image segment contains too many image contents for a receiver to present, the receiver should take the same action as if too many image objects were specified on a page.
4. If a receiver supports multiple image contents, it must support them for any type of image. For example, such a receiver must process multiple image contents containing FS10 data without raising an exception, even though FS10 definition specifies a single image content in each image segment.
5. Multiple image contents are treated by the receiver as if they were sent as multiple image objects, in the same order in which they appear in the image segment.
6. All of the image contents are presented using the same image presentation space characteristics, as defined in the image data descriptor for the image object.
7. Function Set 45 is the only current IOCA function set that requires receivers to support multiple image contents in a single image segment.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-0004 **Invalid parameter value**

Condition: The OBJTYPE value is not in the valid range.

EC-910F **Invalid sequence**

Condition: One or more of the following conditions holds:

- A Begin Image Content is missing, or it appeared out of sequence. IOCA receivers can generate an out-of-sequence exception condition, EC-*xx*0F, instead of EC-910F, where *xx* is the one-byte ID code of the IOCA self-defining field encountered in place of the Begin Image Content self-defining field.
- The Begin Image Content has appeared more than once and the receiver supports only a single image content in each image segment.

End Image Content

The End Image Content parameter defines the end of the image content.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'93'	End Image Content	M
1	UBIN	LENGTH	X'00'	Length of the parameters to follow	M

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

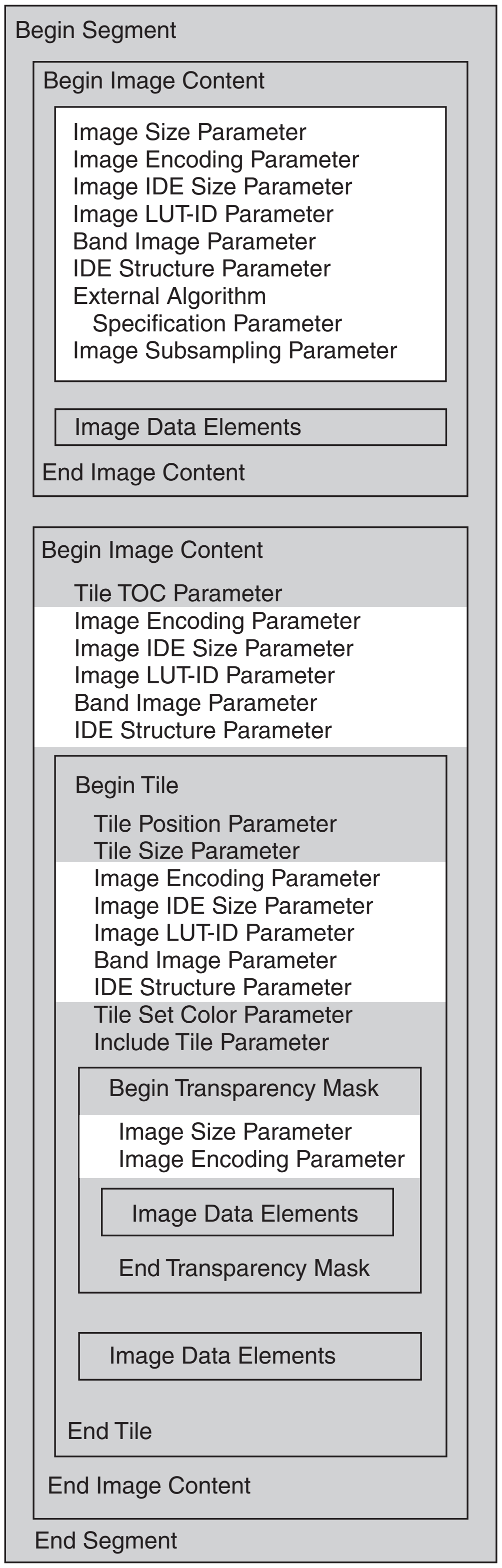
Condition: The LENGTH value is not in the valid range.

EC-930F **Invalid sequence**

| **Condition:** An End Image Content is missing, or it appeared out of sequence. IOCA receivers can generate an out-of-sequence exception condition,, EC-*xx*0F, instead of EC-930F, where *xx* is the one-byte ID code of the IOCA self-defining field encountered in place of the End Image Content self-defining field.

Image data parameters

Image data parameters describe the characteristics of the image data within a particular image content. They do not affect the image data in other image contents.



This section describes:
- Image Size parameter
- Image Encoding parameter
- IDE Size parameter
- Image LUT-ID parameter
- Band Image parameter
- IDE Structure parameter
- External Algorithm Specification parameter
- Image Subsampling parameter

The Image Size parameter must exist in each untiled image content; the other image data parameters are optional. The Image Size parameter must not exist in a tiled image content. Some optional parameters are not permitted in some *function sets*. If you omit an optional parameter permissible in the function set, its default value is used.

In a tiled image content, the image data parameters described in this section can appear either within tiles or before the first tile. Any value set in an image data parameter specified before the first tile is used as a default in all the tiles. The same image data parameter can appear outside of tiles and within a tile, in which case the values specified within the tile are used.

A function set is a set of self-defining fields that describes an image object. For more information on function sets, see “Function sets” on page 91.

Image Size

| This self-defining field, which is mandatory in non-tiled image contents, describes
| the measurement characteristics of the image when it is created. There is no default
| value.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'94'	Image Size parameter	M
1	UBIN	LENGTH	X'09'	Length of the parameters to follow	M
2	CODE	UNITBASE	X'00' – X'02'	Unit base: **X'00'** 10 inches **X'01'** 10 centimeters **X'02'** Logical (resolution ratio) All other values are reserved.	M
3–4	UBIN	HRESOL	X'0000' – X'7FFF'	Horizontal resolution	M
5–6	UBIN	VRESOL	X'0000' – X'7FFF'	Vertical resolution	M
7–8	UBIN	HSIZE	X'0000' – X'7FFF'	Horizontal size in image points (excluding any padding bit in each scan line)	M
9–10	UBIN	VSIZE	X'0000' – X'7FFF'	Vertical size in image points (excluding any padding scan line)	M

UNITBASE=X'02' (logical) indicates that the following HRESOL and VRESOL specify a ratio of the horizontal and vertical resolutions.

The combinations of UNITBASE, HRESOL, and VRESOL have the following meanings:

- When UNITBASE=X'00' or X'01':
  - When HRESOL or VRESOL (or both) is zero, the resolution of the image content in that direction is *undefined*. Image contents with undefined resolutions are written with each image point mapped onto one point in the image presentation space.
  - Nonzero HRESOL or VRESOL values, divided by 10, yield the number of image points per inch or per centimeter in the corresponding direction.

    Example: If the distance between image points is 1/200th of an inch, the resolutions are specified as X'0007D007D0'. This means that there are 2000 image points per 10 inches in both the horizontal and vertical directions.
- With UNITBASE=X'02':
  - When either HRESOL or VRESOL is zero, the image content's resolutions in both directions are undefined. Image contents with undefined resolutions are written with each image point mapped on a point in the image presentation space.

– Dividing a nonzero HRESOL value by a nonzero VRESOL value yields the ratio of the horizontal and vertical resolutions.

Example: X'0200010002' means that the vertical resolution is twice the horizontal resolution, and that the image is sharper in the vertical direction than in the horizontal direction. To keep this ratio, the controlling environment allows you to define the image presentation space so as to have the doubled resolution in the vertical direction.

The total number of image points, excluding any padding bit and padding scan line, in the image data can be obtained by multiplying the nonzero HSIZE and VSIZE values.

For non-tiled images, HSIZE=X'00' means that the image data has an *unknown* horizontal size, and VSIZE=X'00' means that it has an unknown vertical size. These are valid only for compression algorithms where the IOCA process model can determine the width or height of the image from the image data during decompression time.

Note: The width or height determined by the IOCA process model may be larger than the actual image width or height, as the image data may include padding bits or padding scan lines.

HSIZE=X'00' or VSIZE=X'00' for other compression algorithms raises exception condition EC-9411. See Appendix A, “Compression and recording algorithms,” on page 129 for details.

When VSIZE=X'00', the actual vertical size of such image data is determined after all image data is received. For example, with InfoPrint MMR—Modified Modified Read the vertical size is determined when the end-of-page (EOP) condition is detected. See Appendix A, “Compression and recording algorithms,” on page 129 for details.

Note: IOCA generators should set HSIZE and VSIZE to the image’s actual width and height regardless of the compression algorithm used. Leaving either HSIZE or VSIZE to zero may cause some IOCA receivers to abort prematurely.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-0004 **Invalid parameter value**

Condition: The HRESOL, VRESOL, HSIZE, or VSIZE value is not in the valid range.

EC-940F **Invalid sequence**

Condition: An Image Size parameter is missing, or it appeared out of sequence or more than once.

Image Size

EC-9410 Invalid or unsupported Image Data parameter value

Condition: The Image Size parameter contains an invalid or unsupported value.

EC-9411 Inconsistent Image Data parameters, or inconsistent Image Data parameter and Image Data

Condition: HSIZE or VSIZE is zero (X'0000'), and the size in that direction cannot be determined from the image data.

The following exception condition causes a unique action to be taken:

EC-9401 Inconsistent Image Size parameter value and Image Data

Condition: The size detected in the image data is different from the HSIZE or VSIZE value of the Image Size parameter.

System action: The size detected from the image data is used.

Image Encoding

This optional self-defining field describes the algorithms by which the image data is encoded. See Appendix A, “Compression and recording algorithms,” on page 129 for details.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'95'	Image Encoding parameter	M
1	UBIN	LENGTH	X'02' – X'03'	Length of the parameters to follow	M

Offset	Type	Name	Range	Meaning	M/O
2	CODE	COMPRID	X'00' – X'0D', X'80' – X'84', X'A0' – X'AF'	Compression algorithm: **X'01'** InfoPrint MMR—Modified Modified Read **X'03'** No compression **X'06'** RL4 (Run Length 4) **X'08'** ABIC (Bilevel Q-Coder) **X'09'** TIFF algorithm 2 **X'0A'** Concatenated ABIC **X'0B'** Color compression used by OS/2® Image Support, part number 49F4608 **X'0C'** TIFF PackBits **X'0D'** TIFF LZW **X'20'** Solid Fill Rectangle **X'80'** G3 MH—Modified Huffman (ITU–TSS T.4 Group 3 one-dimensional coding standard for facsimile) **X'81'** G3 MR—Modified READ (ITU–TSS T.4 Group 3 two-dimensional coding option for facsimile) **X'82'** G4 MMR—Modified Modified READ (ITU–TSS T.6 Group 4 two-dimensional coding standard for facsimile) **X'83'** JPEG algorithms (See the External Algorithm Specification parameter for detail) **X'84'** JBIG2 **X'FE'** User-defined algorithms (see the External Algorithm Specification parameter for details) All other values are reserved.	M

Offset	Type	Name	Range	Meaning	M/O
3	CODE	RECID	X'00' – X'04', X'FE'	Recording algorithm: **X'00'** 3800 **X'01'** RIDIC (Recording Image Data Inline Coding) **X'03'** Bottom-to-Top **X'04'** Unpadded RIDIC **X'FE'** See the External Algorithm Specification parameter for details All other values are reserved.	M
4	CODE	BITORDR	X'00' – X'01'	Bit order within each image data byte: **X'00'** Left-to-right **X'01'** Right-to-left All other values are reserved.	O

Notes:

1. When RECID is X'FE', the External Algorithm Specification parameter must also be present within the same image content, otherwise exception condition EC-9F01 exists.
2. The External Algorithm Specification parameter is no longer required when COMPRID is X'83'. If the decompressor in the receiver fails because the compressed datastream requires a feature unimplemented in the decoder, exception EC-9511 occurs.
3. The Solid Fill Rectangle compression algorithm can be used only within tiled images, for bilevel tiles. Otherwise, exception EC-9510 occurs. This compression algorithm indicates that all the image points in the tile are set to the same color and that the tile does not contain any actual image data.
4. JBIG2 is a toolkit with many different capabilities. The standard recognizes a number of profiles that serve the same function as Function Sets in IOCA. Receivers declaring the JBIG2 support must support at least one JBIG2 profile, but are not obliged to support all of them. If a receiver encounters JBIG2-compressed data encoding unsupported function, exception EC-9511 occurs.
5. LZW encoders sometimes terminate the data early. If the LZW decoder does not produce the expected number of bytes, no exception should be raised and the receiver should fill the remaining data with binary zeroes.

BITORDR indicates the bit order within each image data byte. Figure 14 on page 36, for example, shows a bilevel image with a width of eight image points:

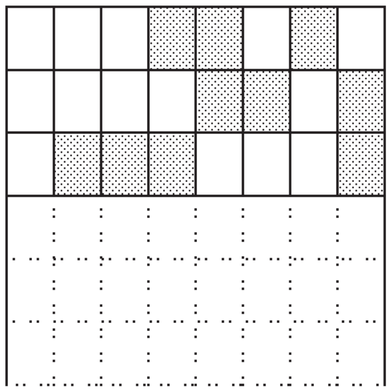


Figure 14. Top three lines of a bilevel image

The uncompressed serial bit stream for the top three lines would be:

```
B'00011010 00001101 01110001 ...'
```

When the bits are packed into image data bytes, with BITORDR=X'00', the first three bytes would be as follows:

```
B'00011010 00001101 01110001 ...'
```

For BITORDR=X'01', the first three bytes of the image data would be:

```
B'01011000 10110000 10001110 ...'
```

If the image data is compressed, the BITORDR parameter denotes the bit order within each compressed image data byte prior to decompression.

Zero is the default for BITORDR if it is absent.

If the Image Encoding parameter is not present, the defaults are X'03' for the compression algorithm, X'01' for the recording algorithm, and zero for the bit order.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 Invalid length

Condition: The LENGTH value is not in the valid range.

EC-0005 Invalid Length

Condition: The LENGTH value is not in the valid, function-set specified range. EC-0005 is optional: IOCA receivers can generate EC-0003 instead of EC-0005.

EC-950F Invalid sequence

Condition: The Image Encoding parameter is required in some function sets but missing, or it appeared out of sequence or more than once.

EC-9510 Invalid or unsupported Image Data parameter value

Condition: The Image Encoding parameter contains an invalid or unsupported value.

The following exception condition causes a unique action to be taken:

EC-9511 Inconsistent Image Data parameters, or inconsistent Image Data parameter and Image Data

Condition: The decoder encountered one of the following conditions when decompressing the image data:

- The image data is not encoded according to the compression or recording algorithm specified in the Image Encoding parameter.
- The image data cannot be decoded successfully using the size values specified in the Image Size parameter. This condition applies to compression or recording algorithms which do not permit the image size to be encoded in the image data.
- The image data is not in complete accordance with the compression algorithm specified in the Image Encoding parameter.
- Image is encoded using the algorithm specified in the Image Encoding Parameter, but uses a function of the algorithm that is unsupported by the receiver.

System action: Receivers should attempt to present or make use of all successfully decompressed image data. Note, however, that the resulting partial image might differ from the original image.

IDE Size

This optional self-defining field specifies the number of bits that comprise each image data element (IDE) in the image data, before any subsampling or compression method is performed on the IDEs.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'96'	IDE Size parameter	M
1	UBIN	LENGTH	X'01'	Length of the parameters to follow	M
2	UBIN	IDESZ	X'01' – X'FF'	Number of bits in each IDE	M

If the IDE Size parameter is not present, the default value for IDESZ is 1 (bilevel image).

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-0004 **Invalid parameter value**

Condition: The IDESZ value is not in the valid range.

EC-960F **Invalid sequence**

Condition: The IDE Size parameter appeared out of sequence or more than once.

EC-9610 **Invalid or unsupported Image Data parameter value**

Condition: The IDE Size parameter contains an invalid or unsupported value.

EC-9611 **Inconsistent Image Data parameters, or inconsistent Image Data parameter and Image Data**

Condition: The compression scheme specified in the Image Encoding parameter does not support the IDE size specified in the IDE Size parameter.

Image LUT-ID

This optional self-defining field identifies the LUT-ID (LUT) that should be used to interpret the image data. Each IDE value is an index into this LUT.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'97'	Image LUT-ID parameter	M
1	UBIN	LENGTH	X'01'	Length of the parameters to follow	M
2	CODE	LUTID	X'00' – X'FF'	LUT-ID identifier	M

If the Image LUT-ID parameter is not present, the default value for LUTID is zero for the standard LUT-ID. See Appendix B, “Bilevel, grayscale, and color images,” on page 139 for details about LUTID.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-970F **Invalid sequence**

Condition: The Image LUT-ID parameter appeared out of sequence or more than once.

EC-9710 **Invalid or unsupported Image Data parameter value**

Condition: The Image LUT-ID parameter contains an invalid or unsupported value.

Band Image

This optional self-defining field describes the format of one or more *bands* that represent an image. A band is a plane where, typically, image data of similar attributes is placed. Certain bits of an IDE can be placed into separate bands, for example the bits that represent the red, green, and blue color components of each IDE.

If the Band Image parameter is present, then the image data must be carried by the Band Image Data self-defining field

Each band of the image IDEs is carried by one or more Band Image Data self-defining fields. The Band Image Data self-defining field is described in "Band Image Data" on page 80.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'98'	Band Image parameter	M
1	UBIN	LENGTH	X'02' – X'FE'	Length of the parameters to follow	M
2	UBIN	BCOUNT	X'01' – X'FD'	Number of bands	M
One or more repeating groups in the following format:					
0	UBIN	BITCNT	X'01' – X'FF'	Bit count for the band	M

BITCNT specifies how many bits of the IDE comprise one band, and BCOUNT specifies how many bands comprise the image data. The number of BITCNTs in the self-defining field must equal the BCOUNT value. The BITCNTs appear in the order in which the bits were placed into the band. For boundary alignment purposes, BITCNT can include padding bits inserted into the data. If BITCNT contains no padding bits, then the sum of all the BITCNT values equals the IDE size specified by the IDE Size parameter.

Example 1: For a single-band image with an IDE size of four with no padding bit, the first four bits of data represent the first IDE, the next four represent the second IDE, and so on.

Figure 15 illustrates the layout of image bits in this image.

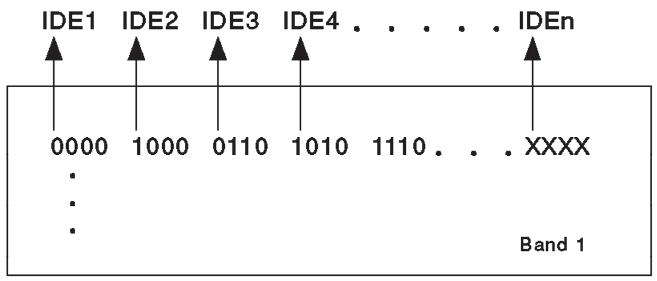


Figure 15. Example of a four-bit single-band Image with No Padding bit

Example 2: For an image with an IDE size of four that is represented by four bands with no padding bit, the first bit in each of the four bands represents the first IDE, the second bit represents the second IDE, and so on.

Figure 16 illustrates the layout of image bits in this image.

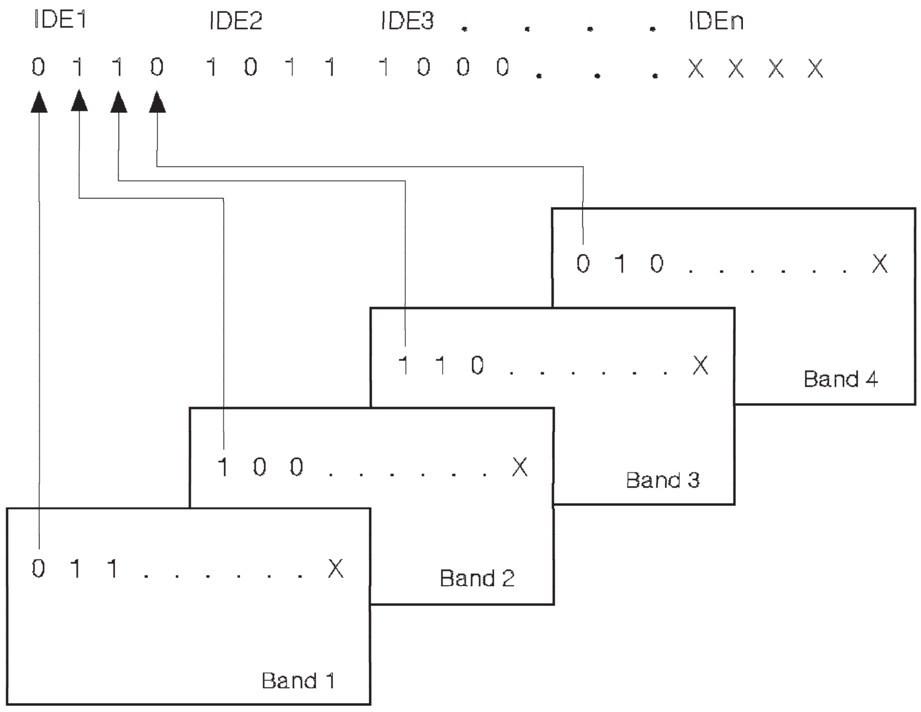


Figure 16. Example of a four-bit four-band Image with No Padding bit

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-0004 **Invalid parameter value**

Condition: The BCOUNT or BITCNT value is not in the valid range.

EC-0005 **Invalid Length**

Condition: The LENGTH value is not in the valid, function-set specified range. EC-0005 is optional: IOCA receivers can generate EC-0003 instead of EC-0005.

EC-9801 **Invalid Band Image parameter and Image Subsampling parameter coexistence**

Condition: In some functions sets, Band Image parameter and Image Subsampling parameter cannot coexist in the same Image Content.

Band Image

EC-980F **Invalid sequence**

Condition: The Band Image parameter appeared out of sequence or more than once.

EC-9810 **Invalid or unsupported Image Data parameter value**

Condition: The Band Image parameter contains an invalid or unsupported value.

EC-9814 **Invalid number of bands and bit counts**

Condition: The number of BITCNT parameters is not equal to the BCOUNT in the Band Image parameter.

EC-9815 **Invalid IDE size**

Condition: The IDE size, determined by the Band Image parameter, does not match the IDE Size parameter.

IDE Structure

This optional self-defining field describes the structure of each IDE for a bilevel, grayscale, or color image, and generates a LUT to interpret the IDEs.

If the IDE Structure parameter is not present, each IDE of the image data consists of a single component whose size is dependent on the IDE Size parameter. The IDE is then used as an index into either the LUT specified by the Image LUT-ID parameter, or the standard LUT.

With this self-defining field, color images are expressed by using the RGB, YCrCb, YCbCr or CMYK model, while grayscale images are expressed by using only the Y component of the YCrCb or YCbCr model.

See Appendix B, “Bilevel, grayscale, and color images,” on page 139 for details on the relationship with the IDE Size parameter and the Image LUT-ID parameter.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'9B'	IDE Structure parameter	M
1	UBIN	LENGTH	X'06' – X'09'	Length of the parameters to follow	M
2	BITS	FLAGS			M
Bit 0		ASFLAG	B'0' – B'1'	Additive or Subtractive: **B'0'** Additive **B'1'** Subtractive	
Bit 1		GRAYCODE	B'0' – B'1'	Gray coding: **B'0'** Off **B'1'** On	
Bits 2–7			B'000000'	Reserved; should be zero	
3	CODE	FORMAT	X'01' – X'02', X'04', X'12'	Color model: **X'01'** RGB **X'02'** YCrCb **X'12'** YCbCr All other values are reserved.	M
4–6			X'000000'	Reserved; should be zero	M
7	UBIN	SIZE1	X'00' – X'FF'	Number of bits/IDE for component 1	M
8	UBIN	SIZE2	X'00' – X'FF'	Number of bits/IDE for component 2	O
9	UBIN	SIZE3	X'00' – X'FF'	Number of bits/IDE for component 3	O
10	UBIN	SIZE4	X'00' – X'FF'	Number of bits/IDE for component 4	O

You can specify whether increasing IDE values correspond to brighter or darker levels of gray or color, with the ASFLAG=0 (*additive*) or ASFLAG=1 (*subtractive*) parameters, respectively. ASFLAG applies to all three components of the RGB model, all four components of the CMYK model, and to the Y component of the YCrCb and YCbCr models.

- *Additive* means that the maximum color value represents full intensity of that color, while the minimum color value represents zero intensity. For example, in a black-and-white system, the minimum color value (usually zero) means black, and the maximum value means white.
- *Subtractive* means that the minimum color value represents full intensity of that color, while the maximum color value represents zero intensity. For example, in a black-and-white system, the minimum color value (usually zero) means white, and the maximum value means black.

FORMAT specifies the breakdown format for each IDE value:

- RGB means that each value is to be treated as a set of red, green, blue intensity values, and the set is in the order red, green, blue.
- YCrCb means that each value is to be treated as a set of Y, Cr, Cb values, and the set is in the order Y, Cr, Cb, where Y is the intensity, and Cr and Cb are the chrominance differences.
- YCbCr means that each value is to be treated as a set of Y, Cb, Cr values, and the set is in the order Y, Cb, Cr, where Y is the intensity, and Cb and Cr are the chrominance differences.
- CMYK means that each value is to be treated as a set of cyan, magenta, yellow, black intensity values and the set is in the order cyan, magenta, yellow, black.

GRAYCODE specifies whether or not the Gray coding scheme is used to encode the image data. Gray code is a type of binary code that is applied to the entire IDE whose size is specified in the IDE Size parameter self-defining field, not just to each individual bit plane of the IDE. Gray code is constructed such that two successive codes always differ by just one bit. Table 2[1] shows the series of gray codes from 0 to 15 in decimal.

Table 2. Gray code values (decimal)

Decimal	**Gray code**
0	B'0000'
1	B'0001'
2	B'0011'
3	B'0010'
4	B'0110'
5	B'0111'
6	B'0101'
7	B'0100'
8	B'1100'
9	B'1101'
10	B'1111'
11	B'1110'
12	B'1010'
13	B'1011'
14	B'1001'
15	B'1000'

1. Source: R. W. Lucky, J. Salz, and E. J. Weldon Jr., *Principles of Data Communication* (New York: McGraw-Hill, 1968).

Refer to R. Hunt, *The Representation of Colour in Photography, Printing and Television* (Foundation Press, 1995), for an explanation of each color model.

Note: For the interchange environment, color images can be expressed by the RGB, YCrCb, or YCbCr color model; however, only RGB images can reference an external LUT. Refer to the resource appendix in the *Mixed Object Document Content Architecture Reference* for more details.

SIZE1, SIZE2, SIZE3, and SIZE4 specify the number of bits required to express each color component of an IDE before any subsampling or compression method is performed on the IDEs. The maximum possible value of a particular color component is equal to $2^{\mathrm{SIZE}n}-1$, where n is 1, 2, 3, or 4.

SIZE1, SIZE2, SIZE3, and SIZE4 must appear in the sequence of the color components whose size they specify. For an RGB image, this sequence is R, G, and B; for a YCrCb image, it is Y, Cr, and Cb; for a YCbCr image, it is Y, Cb, and Cr and for a CMYK image, it is C, M, Y and K.

The number of SIZE parameters varies from one to four, depending on the color components that are used to express each IDE.

For grayscale images, expressed by the YCrCb or YCbCr color model, specifying SIZE1 is sufficient; SIZE2 and SIZE3 can be omitted, or you can specify zero for them. However, any preceding SIZE parameter must be included, and zero must be specified. For example, if an image uses only the third component of a color model, then SIZE1=0 and SIZE2=0 must be specified.

The SIZE4 field is only allowed only for the CMYK color space (IDE Size of 4 or 32), where it is mandatory. If SIZE4 is missing for the CMYK color space, or if it appears for any other color space, exception EC-9B18 occurs.

For the CMYK color space, the color value is specified with four components. Components 1, 2, 3 and 4 are unsigned binary numbers that specify the cyan, magenta, yellow, and black intensity values, in that order. SIZE1, SIZE2, SIZE3 and SIZE4 in the IDE Structure parameter are non-zero and define the number of bits used in each component. The intensity range for the C, M, Y, K components is 0 to 1, which is mapped to the binary value range 0 to $2^{\mathrm{SIZE}n}$-1, where n=1,2,3,4. This is a device-dependent color space.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 The LENGTH value is not in the valid range

Condition: The LENGTH value is not in the valid range.

EC-0005 Invalid Length

Condition: The LENGTH value is not in the valid, function-set specified range. EC-0005 is optional: IOCA receivers can generate EC-0003 instead of EC-0005.

EC-9B0F Invalid sequence

Condition: The IDE Structure parameter is required but missing, or it appeared out of sequence or more than once.

IDE Structure

EC-9B10 Invalid or unsupported Image Data parameter value

Condition: The IDE Structure parameter contains an invalid or unsupported value.

EC-9B18 Invalid IDE Structure parameter

| **Condition:** One of the following conditions has been encountered:

| • The sum of SIZE1 through SIZE4 does not match the IDE size specified by the IDE Size parameter.

| • Color space is CMYK and SIZE4 is missing.

| • SIZE4 is present and the color space is not CMYK.

External Algorithm Specification

This optional self-defining field provides complementary information about the algorithm specified in the Image Encoding parameter. It can be used only in conjunction with that parameter.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'9F'	External Algorithm Specification parameter	M
1	UBIN	LENGTH	X'03' – X'FF'	Length of the parameters to follow	M
2	CODE	ALGTYPE	X'00', X'10'	Type of algorithm specified: **X'00'** Recording algorithm **X'10'** Compression algorithm All other values are reserved.	M
3			X'00F'	Reserved; should be zero	M
4–*n*	CODE	ALGSPEC		Recording Algorithm Specification or Compression Algorithm Specification	M

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-0005 **Invalid Length**

Condition: The LENGTH value is not in the valid, function-set specified range. EC-0005 is optional: IOCA receivers can generate EC-0003 instead of EC-0005.

EC-9F01 **Missing External Algorithm Specification parameter or Image Encoding parameter**

Condition: An External Algorithm Specification parameter exists without a corresponding Image Encoding parameter, or an Image Encoding parameter exists that requires an External Algorithm Specification parameter that cannot be found.

EC-9F0F **Invalid sequence**

Condition: An External Algorithm Specification parameter appeared out of sequence.

EC-9F10 **Invalid or unsupported Image Data parameter value**

Condition: The External Algorithm Specification parameter contains an invalid or unsupported value.

EC-9F11 **Inconsistent Image Data parameters, or inconsistent Image Data parameter and Image Data**

Condition: An External Algorithm Specification parameter is present, but the Image Encoding parameter does not require it.

Recording Algorithm Specification

| This subparameter is carried by the External Algorithm Specification parameter.

Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	BITS	DIRCTN		Direction of IDEs	M
Bits 0–1		IDEPTH	B'11'	Direction of successive IDEs along a line (clockwise from X-axis): **B'11'** 270 degrees All other values are reserved.	
Bits 2–3		LINEPRG	B'00'	Direction of progression of successive lines (clockwise from X-axis): **B'00'** 0 degrees All other values are reserved.	
Bit 4		RNDTRIP	B'0'	Direction of successive IDEs relative to the previous line (clockwise from the previous line): **B'0'** 0 degrees All other values are reserved.	
Bits 5–7		RSVD	B'000'	Reserved; should be zero	
1	CODE	PADBDRY	X'03'	Boundary length for padding: **X'03'** 32-bit boundary All other values are reserved.	M
2	CODE	PADALMT	X'00'	Alignment for padding: **X'00'** Data left-aligned within boundary All other values are reserved.	M

DIRCTN specifies how the IDEs are positioned in a set of image data self-defining fields. The following subparameters are defined:

- IDEPTH specifies how successive IDEs proceed along a line in relation to the X-axis of the image coordinate system. Degrees are measured clockwise from the X-axis of the image coordinate system.
- LINEPRG specifies how successive lines of IDEs proceed in relation to the X-axis of the image coordinate system. Degrees are measured clockwise from the X-axis of the image coordinate system.
- RNDTRIP specifies how the next line of IDEs proceeds in relation to the previous line. Degrees are measured clockwise from the previous line.

PADBDRY specifies if each line of the IDEs is padded with zeros where necessary for boundary alignment purposes.

PADALMT specifies whether the padding bits used for alignment purposes are located at the beginning or at the end of each line of the IDEs.

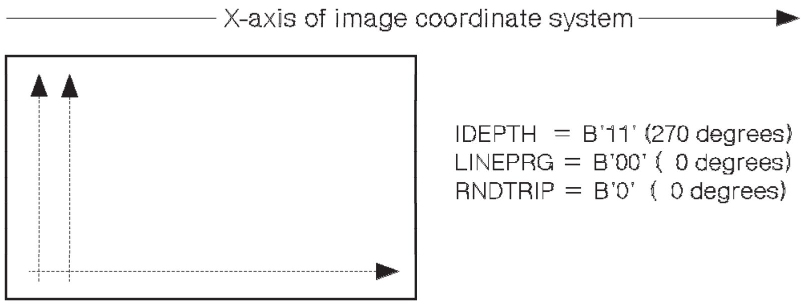

X-axis of image coordinate system
IDEPTH = B'11' (270 degrees)
LINEPRG = B'00' (0 degrees)
RNDTRIP = B'0' (0 degrees)

Figure 17. IDE progression

Exception conditions: The following exception condition causes the standard action to be taken:

EC-9F10 **Invalid or unsupported Image Data parameter value**

Condition: The External Algorithm Specification parameter contains an invalid or unsupported value.

Compression Algorithm Specification

This subparameter is carried by the External Algorithm Specification parameter. The syntax table specifies the JPEG compression algorithm that conforms to the following publications:

- ITU–TSS Recommendation T.81
- ISO/IEC International Standard 10918-1

Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	CODE	COMPRID	X'83'	JPEG algorithms	M
1			X'00'	Reserved; should be zero	M
2	CODE	VERSION	X'00'	Version	M
3			X'00'	Reserved; should be zero	M
4	CODE	MARKER	X'C0' – X'C3', X'C5' – X'C7', X'C9' – X'CB', X'CD' – X'CF'	Marker code: *Non-differential Huffman coding:* **X'C0'** Baseline DCT **X'C1'** Extended sequential DCT **X'C2'** Progressive DCT **X'C3'** Lossless (sequential) *Differential Huffman coding:* **X'C5'** Differential sequential DCT **X'C6'** Differential progressive DCT **X'C7'** Differential lossless *Non-differential arithmetic coding:* **X'C9'** Extended sequential DCT **X'CA'** Progressive DCT **X'CB'** Lossless (sequential) *Differential arithmetic coding:* **X'CD'** Differential sequential DCT **X'CE'** Differential progressive DCT **X'CF'** Differential lossless All other values are reserved.	M
5–7		RESERVED	X'000000'	Reserved; should be zero	M

JPEG algorithms have the following restrictions:

- They cannot be applied to images whose IDE size is 1 bit/IDE;
- The baseline DCT-based algorithm is applicable only to images with 8-bit/component. The other DCT-based algorithms are applicable only to images with 8 bits per component or 12 bits per component. The IDE of the image can consist of at most four components.
- The lossless algorithms are applicable only to images with n bits per component, where $2 \leq n \leq 16$. The IDE of the image can consist of at most four components.

Exception conditions: The following exception condition causes the standard action to be taken:

EC-9F10 **Invalid or unsupported Image Data parameter value**

Condition: The External Algorithm Specification parameter contains an invalid or unsupported value.

Syntax of a user-defined compression algorithm:

Offset	Type	Name	Range	Meaning	M/O
0	CODE	COMPRID	X'FE'	User-defined compression algorithm	M
1	UBIN	LENGTH	X'04' – X'FF'	Length of the parameters to follow	M
2–5	CODE	USRCPID		Architecture-assigned user compression algorithm code point The assignment of compression code points is controlled by the IOCA data stream architecture group.	M
6–*n*		COMPSPEC	Any	User-defined specification	O

Image Subsampling

This optional self-defining field describes the *subsampling* methods used to encode the uncompressed IDEs within the image data. The methods are encoded in self-defining fields.

Subsampling is a technique of reducing the amount of image data, resulting in lower storage and processing requirements. This is accomplished by combining the color information of adjacent IDEs. If done properly, there is little or no visual degradation of the image quality.

Subsampling relies on the fact that in color images the difference between adjacent IDEs is small for certain color components. For example, in the YCrCb and YCbCr color models, most of the image information is concentrated in the Y component; hence it is fairly common to store only the average values of the Cr and Cb components of two adjacent IDEs.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0–1	CODE	ID	X'FECE'	Image Subsampling parameter	M
2–3	UBIN	LENGTH	X'0000' – X'FFFF'	Length of the parameters to follow	M
4–*n*	CODE	SDF		Zero or more self-defining fields that specify the subsampling methods	O

If the Image Subsampling parameter is not present, the default is that the IDEs have not been subsampled.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 The LENGTH value is not in the valid range

Condition: The LENGTH value is not in the valid range.

EC-0004 Invalid parameter value

Condition: The HSAMPLE or VSAMPLE value is not in the valid range.

EC-0005 Invalid Length

Condition: The LENGTH value is not in the valid, function-set specified range. EC-0005 is optional: IOCA receivers can generate EC-0003 o EC0005.

EC-CE01 Invalid Band Image parameter and Image Subsampling parameter coexistence

Condition: In some function sets, Band Image parameter and Subsamploing parameter cannot coexist in the same Image Content.

EC-CE0F **Invalid sequence**

Condition: The Image Subsampling parameter appeared out of sequence or more than once.

EC-CE10 **Invalid or unsupported Image Data parameter value**

Condition: The Image Subsampling parameter contains an invalid or unsupported value.

Sampling Ratios

This optional self-defining field is carried by the Image Subsampling parameter described in “Image Subsampling” on page 52. It specifies the number of horizontal and vertical samples that make up each component of the IDEs.

Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'01'	Sampling Ratios	M
1	UBIN	LENGTH	X'02' – X'FE'	Length of the parameters to follow	M
1 to 127 repeating groups in the following format:					
0	UBIN	HSAMPLE	X'00' – X'7F'	Number of horizontal samples that make up the component of the IDEs	M
1	UBIN	VSAMPLE	X'00' – X'7F'	Number of vertical samples that make up the component of the IDEs	M

If the HSAMPLE and VSAMPLE group for a particular component of the IDEs is not present, the default value is 1 for both HSAMPLE and VSAMPLE. However, any preceding HSAMPLE and VSAMPLE group must be included. For example, a color image with only its third component subsampled must have HSAMPLE1, VSAMPLE1, HSAMPLE2, and VSAMPLE2 specified as equal to 1.

Example: For a 24-bit YCrCb uncompressed color image that has eight bits per component using the following sampling ratios:

```
HSAMPLE1=2    VSAMPLE1=1
HSAMPLE2=1    VSAMPLE2=1
HSAMPLE3=1    VSAMPLE3=1
```

the resulting image data layout would be as follows:

Offset	**Content**
0	The Y component value of the first IDE
1	The Y component value of the second IDE
2	The average of the first and second IDEs Cr component values
3	The average of the first and second IDEs Cb component values
4	The Y component value of the third IDE
5	The Y component value of the fourth IDE
6	The average of the third and fourth IDEs Cr component values
7	The average of the third and fourth IDEs Cb component values
etc.	etc.

Exception conditions: The following exception conditions cause the standard action to be taken:

EC-0003 **The LENGTH value is not in the valid range**

Condition: The LENGTH value is not in the valid range.

EC-0004 **Invalid parameter value**

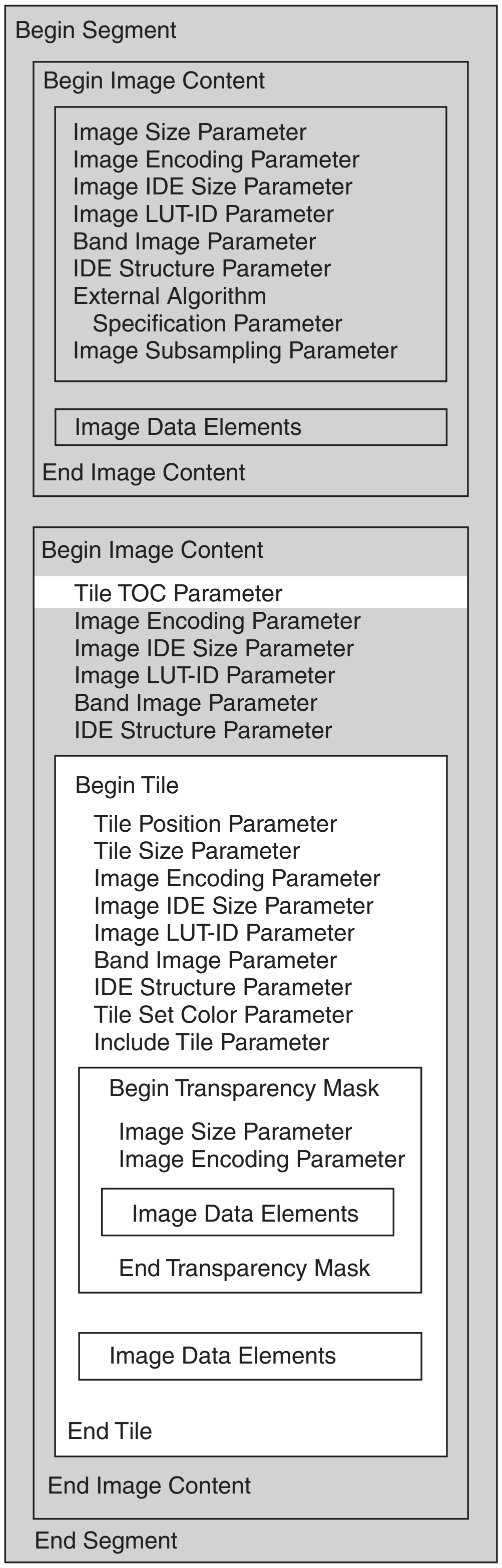
Condition: The HSAMPLE or VSAMPLE value is not in the valid range.

EC-0005 **Invalid Length**

Condition: The LENGTH value is not in the valid, function-set specified range. EC-0005 is optional: IOCA receivers can generate EC-0003 instead of EC-0005.

Tiles

Tiles are used when different parts of an image are described using different colorspaces, resolutions and compression algorithms. Tiles can also be used as resources (see Appendix C, “IOCA Tile Resource,” on page 143).



Tiles

The tiling scheme used in IOCA has the following features:

- Each image content can be either tiled or untiled. In an untiled image content, no tiles may appear. Tiled image contents are indicated by the presence of the Tile TOC parameter immediately following the Begin Image Content Parameter. In a tiled image content, no image data elements may appear outside of the tiles.
- Tiles can use different color spaces and compression algorithms.
- Each tile must either have the resolution of the underlying image presentation space, or be subsampled by the same integer factor in both horizontal and vertical dimensions.
- Tiles must be non-overlapping and must also be specified in top-down, left-to-right order.
- Tiles do not have to cover the whole image presentation space. The part of the image presentation space not covered by tiles is treated as background. Tiles must be fully contained in the image presentation space.
- Within tiles, foreground and background are determined based on the color space used.
- A tile can be either a *data tile* (that is, a fully defined tile with all the data present), or a *referencing tile*. A referencing tile contains an invocation, positioning and merging instruction for a tile resource and is intended to save bandwidth and processing time when processing multiple images that have some areas in common.

Begin Tile

The Begin Tile parameter defines the beginning of a tile.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'8C'	Begin Tile	M
1	UBIN	LENGTH	X'00'	Length of the parameters to follow	M

Notes:

1. In tiled images, all of the image data must be contained in tiles. That is, no image data or band image data can appear outside of the sequence delimited by the Begin Tile/End Tile pairs.
2. The Begin Tile Parameter can appear in all of the contexts where image data and band image data can appear in non-tiled images.
3. If Begin Tile Parameter is encountered, the first parameter after the Begin Image Content parameter must be the Tile TOC parameter. Otherwise, exception EC-8C0F occurs.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-8C0F **Invalid sequence**

Condition: A Begin Tile has appeared out of sequence.

End Tile

The End Tile parameter defines the end of a tile.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'8D'	End Tile	M
1	UBIN	LENGTH	X'00'	Length of the parameters to follow	M

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-8D0F **Invalid sequence**

Condition: An End Tile is missing after a Begin Tile has been encountered, or it appeared out of sequence.

Tile Position

The Tile Position parameter determines the position of the left upper corner of the tile in the image presentation space.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'B5'	Tile Position	M
	UBIN	LENGTH	X'08'	Length of the parameters to follow	M
2–5	UBIN	XOFFSET	X'00000000' – X'7FFFFFFF'	Horizontal offset of the tile origin, relative to the presentation space origin	M
6–9	UBIN	YOFFSET	X'00000000' – X'7FFFFFFF'	Vertical offset of the tile origin, relative to the presentation space origin	M

Notes:

1. The XOFFSET and YOFFSET are specified in the presentation space image points. If subsampling is specified in the Tile Size Parameter, it does not apply to the XOFFSET and YOFFSET.
2. The left upper corner of the tile must be contained in the presentation space; that is, the XOFFSET and YOFFSET must be less than XSIZE and YSIZE respectively, as specified in the Image Data Descriptor. For the definition of the Image Data Descriptor, see “Image Data Descriptor (IDD)” on page 146.
3. If the current tile is not the first tile specified, the YOFFSET value must be at least as large as any specified for the previous tiles. If YOFFSET is identical to the previous YOFFSET, XOFFSET must be greater than the previous XOFFSET. This requirement forces the tile order of top down (primary key) and left to right (secondary key). This condition applies only if the Tile TOC Parameter does not contain the tile table of contents.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-B50F **Invalid sequence**

Condition: A Tile Position is missing, or it appeared out of sequence.

EC-B510 **Invalid Tile Position Parameters**

Condition: The XOFFSET, YOFFSET, or both are outside of the valid range or outside of the image presentation space.

EC-B511 **Inconsistent Tile Position Parameters**

Condition: One of the following conditions has been encountered:

- Tiles are specified out of order. This exception can occur only if the Tile TOC Parameter does not contain the table of contents. If the Tile TOC Parameter does contain the table of contents, the tiles themselves can be specified in any order.

- Offset mismatch: the tile table of contents has been specified, but the XOFFSET or YOFFSET given for this tile does not match the values specified in the Tile Position Parameter.

Tile Size

The Tile Size parameter defines the size and resolution of a tile.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'B6'	Tile Size	M
1	UBIN	LENGTH	X'08' – X'09'	Length of the parameters to follow	M
2–5	UBIN	THSIZE	X'00000000' – X'7FFFFFFF'	Horizontal size in image points, excluding any padding bits in each scan line	M
6–9	UBIN	TVSIZE	X'00000000' – X'7FFFFFFF'	Vertical size in image points, excluding any padding scan lines	M
10	UBIN	RELRES	X'01' – X'02'	Relative resolution of the tile.	O

Notes:

1. If RELRES has not been specified, the tile resolution is the same as the resolution of the Image Presentation Space.
2. A RELRES value of 1 means that the tile has the same resolution as the Image Presentation Space. A RELRES value of 2 means the resolution of the tile is half the resolution of the Image Presentation Space. For example, if the Image Presentation Space has a resolution of 600 dpi, a tile with a RELRES of 2 has a resolution of 300 dpi. The default value of RELRES is 1.
3. The tile dimensions THSIZE and TVSIZE are specified in the tile resolution. To get the size of the tile in the presentation space points, multiply the THSIZE and TVSIZE by RELRES.
4. The tile must be wholly contained in the presentation space; that is, (XOFFSET + RELRES × THSIZE) must not exceed the XSIZE specified in the Image Data Descriptor and (YOFFSET + RELRES × THSIZE) must not exceed the YSIZE specified in the Image Data Descriptor.
5. Tiles must be non-overlapping. If X1, Y1, H1, V1, S1 and X2, Y2, H2, V2, S2 describe the offset, size and subsampling of any two tiles, at least one of the following relationships must hold:
   ```
   X1 + S1 × H1 ≤ X2
   X2 + S2 × H2 ≤ X1
   Y1 + S1 × V1 ≤ Y2
   Y2 + S2 × V2 ≤ Y1
   ```

   Note that, in this example, tiles 1 and 2 are not necessarily sorted. That is, the origin of tile 1 need not be above or left of the origin of tile 2.
6. The JPEG compression algorithm works on 8-by-8-pixel blocks. Depending on the JPEG subsampling (note that this is different from RELRES in Tile Size), the Minimum Coded Units (MCUs) used by JPEG might be larger. The most common MCU size is 16 by 16 pixels. The image must be padded before compression to the MCU boundary and the decompressor discards the padding pixels. To help receivers merge JPEG-compressed tiles efficiently, the tile data must be padded to the left and top to the nearest 8-pixel boundary in the tile resolution, after applying tile subsampling and before compression. After padding on the left and top, the tile is padded as usual on the right and

bottom. On decompression, the decompressor discards the right and bottom padding pixels. The receiver then must discard any left and top padding pixels. The number of pad pixels on the left and top can be computed by dividing the XOFFSET and YOFFSET by RELRES×8 and taking the remainder. Note that padding is done in the Image Presentation Space image points, before subsampling. Otherwise, images with odd XOFFSET or YOFFSET could not be aligned.

Example

This example shows how to construct, compress, and decompress a tile with JPEG and RELRES of 2.

Let the area of the image that we wish to use as a tile have the origin of XOFFSET = 21 and YOFFSET = 36. Let the area be 100 presentation space points wide and 211 presentation space points high. Assume that we use no JPEG subsampling. XOFFSET and YOFFSET can be used to indicate the tile origin in the Tile Position parameter. The tile size is be set to THSIZE = 50 and TVSIZE = 105.

To compress the data, start at the image point with the horizontal offset of 16 and the vertical offset of 32 in the presentation space. Select the region 112 pixels wide and 224 pixels high. If the presentation space is not large enough, pad at the right and bottom, until these dimensions are reached. Subsample by the factor of two, which yields an image 56 pixels wide and 112 pixels high. Since the image sizes are even multiples of 8 and no JPEG subsampling is desired, the data can be compressed with JPEG without further padding.

To merge the tile into the presentation space, decompress the tile with JPEG. Upsample by a factor of two, yielding a tile that is 112 by 224 pixels. Since XOFFSET is 21, we know that the leftmost five pixels have to be discarded. Similarly, the YOFFSET value of 36 indicates a top pad of 4 pixels. From the THSIZE and TVSIZE, after upsampling, the actual tile is 100 pixels wide and 210 pixels high. Thus, left 5 pixels, top 4 pixels, right 7 pixels and bottom 10 pixels are discarded to yield the unpadded tile. Note that a scan line on the bottom was lost due to downsampling.

In this example, the right and bottom are padded before the data is passed to the compressor. If you do not pad first, the compressor does the padding and the decompressor strips it. Manual padding, however, allows control over how the padding is done. If the tiles are constructed so that a single continuous tone image is broken into multiple adjoining tiles, selecting the actual image data for padding eliminates edge artifacts when the tiles are joined.

If the compressor allows the caller to specify the padding data, manual padding is not necessary. Note that manual padding also assumes that the receiver checks the image returned by the decompressor and discards not only the top and left pads, but also the bottom and right pads. The receiver can compute the pad sizes from the values of RELRES, XOFFSET, YOFFSET, THSIZE, and TVSIZE.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-A902 **A portion of the extracted image is written outside the Image Presentation Space**

Condition: The tile is not wholly contained in the image presentation space.

EC-B60F **Invalid sequence**

Condition: A Tile Size is missing, or it appeared out of sequence.

EC-B610 **Invalid Tile Size parameters**

Condition: Tile size or relative resolution are outside valid ranges or are invalid for the function set.

EC-B611 **Inconsistent Tile Size parameters**

Condition: At least one of the following conditions is true:
- The tile overlaps a previously specified tile.
- Subsampling mismatch: the RELRES value in the table of contents does not match the RELRES value in the Tile Size parameter.
- Size mismatch: the THSIZE or TVSIZE specified in the table of contents does not match the corresponding value in the Tile Size parameter.

Tile Set Color

The Tile Set Color parameter specifies the color used to paint significant pels of a bilevel tile.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'B7'	Tile Set Color	M
1	UBIN	LENGTH	X'10'	Length of the parameters to follow	M
2	CODE	CSPACE	X'01', X'04', X'06', X'08', X'40'	Color space: **X'01'** RGB **X'04'** CMYK **X'06'** Highlight color space **X'08'** CIELAB **X'40'** Standard OCA color space	M
3–5	UBIN	RESERVED	X'000000'	Reserved; should be zero	M
6	UBIN	SIZE1	X'08'	Number of bits/IDE for component 1; see color space definitions	M
7	UBIN	SIZE2	X'08'	Number of bits/IDE for component 2; see color space definitions	M
8	UBIN	SIZE3	X'08'	Number of bits/IDE for component 3; see color space definitions	M
9	UBIN	SIZE4	X'00', X'08'	Number of bits/IDE for component 4; see color space definitions	M
10–*n*		Color		Color specification; see “Tile Set Color semantics” for details	M

Notes:

1. The Tile Set Color parameter serves two purposes. One purpose is to define the color of the significant pels in a bilevel tile. The other is to paint the whole tile with the specified color. In the second use, the tile does not contain any image data.
2. If the Tile Set Color Parameter is present, the significant image pels are painted with the specified color. Insignificant image pels are treated according to the rules for bilevel images.
3. If all pels are significant (that is, if the whole tile is to be painted), the compression algorithm must be set to Solid Fill Rectangle. In this case (solid fill), image data and band image data cannot appear, or the exceptions EC-920F and EC-9C0F occur.
4. The Image Encoding parameter and IDE Structure parameter can appear for the tile, but must specify a bilevel image (the IDE size must be 1). The color space given in the IDE Structure parameter must be either YCbCr or YCrCb.

Tile Set Color semantics

CSPACE Is a code that defines the color space and the encoding for the color specification.

Value	Description
X'01'	RGB color space. The color value is specified with three components. Components 1, 2, and 3 are unsigned binary numbers that specify the red, green, and blue intensity values, in that order. SIZE1, SIZE2, and SIZE3 are non-zero and define the number of bits used to specify each component. SIZE4 is reserved and should be set to zero. The intensity range for the R,G,B components is 0 to 1, which is mapped to the binary value range 0 to (2^{SIZEN} – 1), where N=1,2,3. **Architecture note:** The reference white point and the chromaticity coordinates for RGB are defined in SMPTE RP 145-1987, entitled *Color Monitor Colorimetry*, and in RP 37-1969, entitled *Color Temperature for Color Television Studio Monitors*, respectively. The reference white point is commonly known as *Illuminant* D_{6500} or simply *D65*. The R,G,B components are assumed to be gamma-corrected (non-linear) with a gamma of 2.2.
X'04'	CMYK color space. The color value is specified with four components. Components 1, 2, 3, and 4 are unsigned binary numbers that specify the cyan, magenta, yellow, and black intensity values, in that order. SIZE1, SIZE2, SIZE3, and SIZE4 are non-zero and define the number of bits used to specify each component. The intensity range for the C,M,Y,K components is 0 to 1, which is mapped to the binary value range 0 to (2^{SIZEN} – 1), where *N*=1,2,3,4. This is a device-dependent color space.
X'06'	Highlight color space. This color space defines a request for the presentation device to generate a highlight color. The color value is specified with one to three components. Component 1 is a two-byte unsigned binary number that specifies the highlight color number. The first highlight color is assigned X'0001', the second highlight color is assigned X'0002', and so on. The value X'0000'. specifies the presentation device default color. SIZE1 = X'10' and defines the number of bits used to specify component 1. Component 2 is an optional one-byte unsigned binary number that specifies a percent coverage for the specified color. Percent coverage can be any value from 0% to 100% (X'00'–X'64'). The number of distinct values supported is presentation-device dependent. If the coverage is less than 100%, the remaining coverage is achieved with color of medium. SIZE2 = X'00' or X'08' and defines the number of bits used to specify component 2. A value of X'00' indicates that component 2 is not specified in the color value, in which case the architected default for percent coverage is 100%. A value of X'08' indicates that component 2 is specified in the color value.

Component 3 is an optional one-byte unsigned binary number that specifies a percent shading, which is a percentage of black that is to be added to the specified color. Percent shading can be any value from 0% to 100% (X'00'–X'64'). The number of distinct values supported is presentation-device dependent. If percent coverage and percent shading are specified, the effective range for percent shading is 0% to (100-coverage)%. If the sum of percent coverage plus percent shading is less than 100%, the remaining coverage is achieved with color of medium. SIZE3 = X'00' or X'08' and defines the number of bits used to specify component 3. A value of X'00' indicates that component 3 is not specified in the color value, in which case the architected default for percent shading is 0%. A value of X'08' indicates that component 3 is specified in the color value.

Implementation note: The percent shading parameter is currently not supported in AFP environments.

SIZE4 is reserved and should be set to zero. This is a device-dependent color space.

Architecture notes:

1. The color that is rendered when a highlight color is specified is device-dependent. For presentation devices that support colors other than black, highlight color values in the range X'0001' to X'FFFF' may be mapped to any color. For bi-level devices, the color may be simulated with a graphic pattern.
2. If the specified highlight color is “presentation device default”, devices whose default color is black use the percent coverage parameter, which is specified in component 2, to render a percent shading.
3. On printing devices, the color of medium is normally white, in which case a coverage of *n*% results in adding (100−*n*)% white to the specified color, or *tinting* the color with (100−*n*)% white. Display devices may assume the color of medium to always be white and use this algorithm to render the specified coverage.
4. The highlight color space can also specify indexed colors when used in conjunction with a Color Mapping Table (CMT) or an Indexed (IX) Color Management Resource (CMR). When used with an Indexed CMR, component 1 specifies a two-byte value that is the index into the CMR and components 2 and 3 are ignored. Note that when both a CMT and Indexed CMRs are used, the CMT is always accessed first. To preserve compatibility with existing highlight color devices, indexed color values X'0000' to X'00FF' are reserved for existing highlight color applications ad devices. That is, indexed color values in range X'0000' to X'00FF', assuming they are not mapped in a CMT, are mapped directly to highlight colors. Indexed color

values in the range X'0100' to X'FFFF', assuming they are not mapped in a CMT, are used to access Indexed CMRs.

X'08' CIELAB color space. The color value is specified with three components. Components 1, 2, and 3 are binary numbers that specify the L, a, b values, in that order, where L is the luminance and a and b are the chrominance differences. Component 1 specifies the L value as an unsigned binary number; components 2 and 3 specify the a and b values as signed binary numbers. SIZE1, SIZE2, and SIZE3 are non-zero and define the number of bits used to specify each component. SIZE4 is reserved and should be set to zero. The range for the L component is 0 to 100, which is mapped to the binary value range 0 to (2^{SIZE1} – 1). The range for the a and b components is –127 to +127, which is mapped to the binary range –($2^{\text{SIZE}N-1}$ – 1) to +($2^{\text{SIZE}N-1}$ – 1).

For color fidelity, 8-bit encoding should be used for each component, that is, SIZE1, SIZE2, and SIZE3 are set to X'08'. When the recommended 8-bit encoding is used for the a and b components, the range is extended to include –128, which is mapped to the value X'80'. If the encoding is less than 8 bits, treatment of the most negative binary endpoint for the a and b components is device-dependent, and tends to be insignificant because of the quantization error.

Architecture note: The reference white point for CIELAB is known as *D50* and is defined in CIE publication 15-2 entitled *Colorimetry*.

X'40' Standard OCA color space. The color value is specified with one component. Component 1 is an unsigned binary number that specifies a named color using a two-byte value from the Standard OCA Color Value Table. SIZE1 = X'10' and defines the number of bits used to specify component 1. SIZE2, SIZE3, SIZE4 are reserved and should be set to zero. This is a device-dependent color space.

All others
Reserved

SIZE1 Defines the number of bits used to specify the first color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary. For example, if SIZE1 = X'06', the first color component has two padding bits.

SIZE2 Defines the number of bits used to specify the second color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

SIZE3 Defines the number of bits used to specify the third color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

SIZE4 Defines the number of bits used to specify the fourth color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

Color Specifies the color value in the defined format and encoding. Note that the number of bytes specified for this parameter depends on the color space. For example, when using 8 bits per component, an RGB color value is specified with 3 bytes, while a CMYK color value is specified with 4 bytes. If extra bytes are specified, they are ignored as long as the self-defining field length is valid.

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

- Item 3 in exception EC-B710 (Invalid Tile Set Color Parameter) will be changed to read “Invalid Color values” instead of “Invalid CVAL values”.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-B70F **Invalid sequence**

Condition: The Tile Set Color parameter appears out of sequence, or more than once within a single tile.

EC-B710 **Invalid Tile Set Color parameter**

Condition: At least one of the following values is not valid:
- CSPACE
- SIZE
- Color

EC-B711 **Inconsistent Tile Set Color parameter:**

Condition: The IDESZ field in the IDE Size parameter has a value other than 1; or the colorspace specified in the IDE Structure parameter is not YCbCr or YCrCb.

Include Tile

The Include Tile parameter defines the tile as referencing tile. The tile does not contain any image data, except possibly a transparency mask, and is instead read from the referenced resource.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0–1	CODE	ID	X'FEB8'	Include Tile	M
2–3	UBIN	LENGTH	X'0004'	Length of the parameters to follow	M
4–7	CODE	TIRID	X'00000000' – X'FFFFFFFF'	Tile Resource local identifier	M

Notes:

1. If a tile contains the Include Tile parameter, it must contain a Tile Position parameter and can also contain a transparency mask. Any other parameters cause one of the Invalid Sequence (EC-XX0F) exceptions to be raised.
2. The Tile Position parameter in the included tile is ignored. The Tile Position parameter specified in the referencing tile is used instead.
3. If a referencing tile contains a transparency mask and the included tile also contains a transparency mask, the two masks are combined by using the logical AND operation. That is, a pixel is foreground if it is defined as foreground in both masks.
4. Tile resources do not contain any references to the image presentation space. Each included tile is interpreted according to the current image presentation space.
5. Except for the Tile Position and transparency mask, the included tile is treated exactly as if it was specified entirely locally. All defaulting and override rules for tile data apply.
6. The included tile must not contain another Include Tile parameter (that is, no nested references are allowed). There are no other constraints on the tile content.
7. Any other errors, such as the tile not being contained in the image presentation space, are treated by raising the same exceptions as if the tile were specified locally.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-B80F **Invalid sequence**

Condition: An Include Tile parameter has appeared out of sequence or more than once.

Include Tile

EC-B811 **Inconsistent Include Tile parameter**

Condition: The included tile resource contains an Include Tile parameter.

Tile TOC

The Tile Table of Contents (TOC) parameter defines the image as a tiled image. Optionally, it also defines the size and position of each tile.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0–1	CODE	ID	X'FEBB'	Tile TOC	M
2–3	UBIN	LENGTH	X'2' – X'7FFF'	Length of the parameters to follow. This value must be a multiple of 26, plus 2	M
4–5	UBIN		X'0000'	Reserved; should be zero	M
Zero or more repeating groups, with tile table of contents. If any TOC entries are present, then an entry for each tile must be present. The groups have the following format:					
0–3	UBIN	XOFFSET	X'00000000' – X'7FFFFFFF'	Horizontal offset of the tile origin, relative to the image origin	M
4–7	UBIN	YOFFSET	X'00000000' – X'7FFFFFFF'	Vertical offset of the tile origin, relative to the image origin	M
8–11	UBIN	THSIZE	X'00000000' – X'7FFFFFFF'	Horizontal size in image points, excluding any padding bits in each scan line	M
12–15	UBIN	TVSIZE	X'00000000' – X'7FFFFFFF'	Vertical size in image points, excluding any padding scan lines	M
16	UBIN	RELRES	X'01' – X'02'	Relative resolution of the tile	M
17	CODE	COMPR		Compression algorithm; see Image Encoding parameter	M
18–25	UBIN	DATAPOS	Any	Offset, in bytes, from the start of the Begin Segment parameter of the current image, to the start of the Begin Tile parameter starting the tile	M

Notes:

1. Tiles in the table of contents must be specified in top-down, left-to-right order. If the table of contents is specified, the tiles themselves can be specified in any order (that is, the order restriction described for the Tile Position parameter is lifted).
2. The Tile TOC parameter must appear immediately after the Begin Image Content parameter; otherwise exception EC-BB0F occurs. If a Begin Tile Parameter is encountered without a Tile TOC Parameter having been specified, exception EC-8C0F occurs.
3. If the image contains the Tile TOC parameter, no image data or band image data may appear outside of the tiles (Begin Tile/End Tile pairs). Otherwise, exceptions EC-9201 (Image Data) or EC-9C01 (Band Image Data) occur.

4. The presence of the Tile TOC parameter does not require that any tiles be actually specified. An empty image (no tiles present) is valid and all the image points are treated as background.
5. In the terms of the DATAPOS, the first byte of the Begin Segment has the offset zero.
6. If the Tile TOC parameter contains the table of contents, the values in the table of contents entry for each tile must match the values specified in the Tile Position parameter and Tile Size parameter for that tile. Otherwise, exceptions EC-B511 and EC-B611 respectively occur when inconsistent values are encountered in the Tile Position parameter and the Tile Size parameter.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 Invalid length

Condition: The LENGTH value is not in the valid range.

EC-BB0F Invalid sequence

Condition: A Tile TOC parameter appeared somewhere other than immediately after the Begin Image Content parameter or appeared more than once.

EC-BB10 Invalid Tile TOC values

Condition: One or more values specified in the Tile TOC Parameter is outside of the valid range.

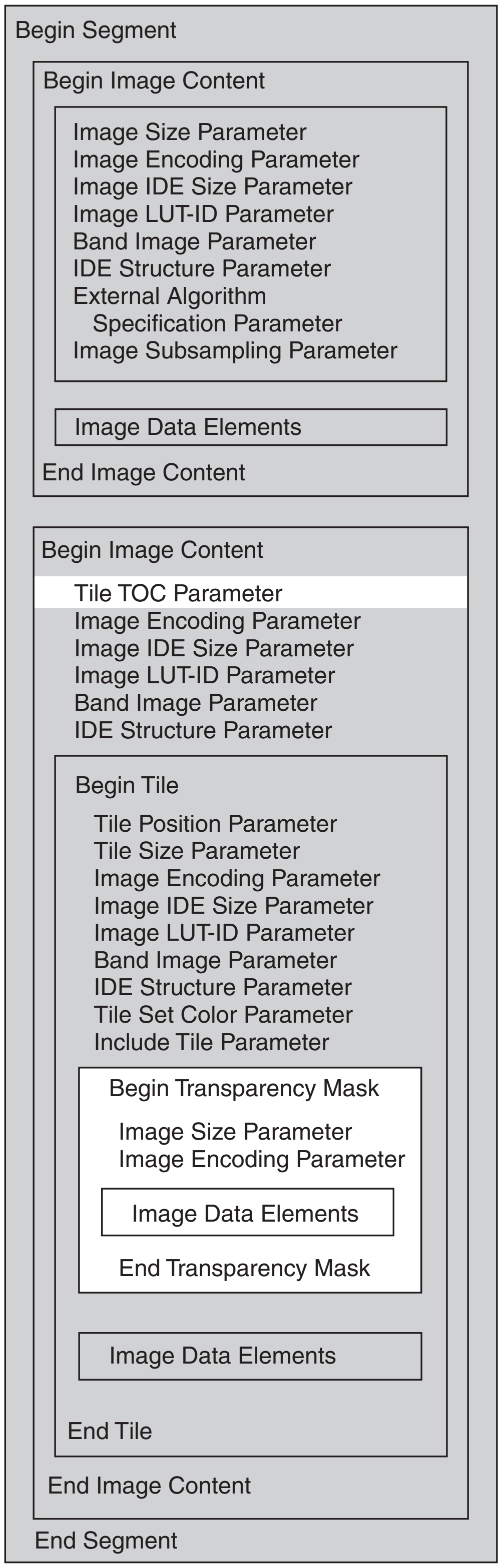
EC-BB11 Inconsistent Tile TOC parameter

Condition: The parameter contains the tile table of contents and one or more of the following conditions is true:
- Not all tiles are listed in the table of contents, even though the table of contents contains at least one tile.
- The table of contents lists a tile that does not exist.
- Invalid tile order: two or more tiles in the table of contents have identical sort keys; or the sort keys are out of sequence.

  Note: The primary sort key is YOFFSET. The secondary sort key is XOFFSET.
- Invalid DATAPOS: the specified offset for one or more tiles does not point to a position where a Begin Tile Parameter starts.

Transparency masks

Transparency masks are bilevel images that are used to turn some image points into background. Function Set 45 is currently the only function set that allows transparency masks. For more information on function sets, see "Function sets" on page 91.



Transparency masks

The transparency mask is a restricted bilevel image in the sense that it must have the same size in pels as the underlying image or tile. If the transparency mask is specified within a tile, the mask has the same resolution as the presentation space; that is, the relative resolution specified in RELRES does not apply. Transparency mask dimensions are carried explicitly using the Image Size parameter. These dimensions must match dimensions obtained by multiplying the tile dimensions by RELRES; otherwise exception EC-9411 occurs.

The transparency mask, if present, must immediately precede the first Image Data or Band Image Data. Images that are not tiled can have at most one transparency mask. In tiled images, the transparency masks must be contained in tiles and each tile can contain at most one transparency mask. Note that tiled images can thus contain multiple transparency masks, each contained in and applying to a different tile.

If the transparency mask is specified in a tile that contains the Include Tile parameter, it must be specified after both the Tile Position and Include Tile parameters.

Tiles using the Include Tile parameter to invoke tile resources can have two transparency masks, one in the calling tile and one in the resource tile itself. The two transparency masks are combined using the logical AND operation; that is, an image point is in the foreground if it is in foreground in both masks. In other words, the caller can declare some of the resource image foreground points as background, but not the reverse.

The transparency mask has a point for each underlying image or presentation space point. If the transparency mask has been specified, the receiver should apply it on a point by point basis. If, at an image point, the mask contains B'0.', the point is treated as background. Otherwise, if the mask contains B'1', the image point is treated according to the rules of the current color space, as if no transparency mask has been specified.

Table 3. Transparency mask structure

	X'8E'	Begin Transparency Mask parameter		
	X'94'	Image Size parameter		
[	X'95'	Image Encoding parameter		]
	X'FE92'	Image Data	(S)	
	X'8F'	End Transparency Mask parameter		

Transparency masks can be described using the following parameters:
- Begin Transparency Mask
- Image Size
- Image Encoding
- Image Data
- End Transparency Mask

Note: All recording algorithms and compression algorithms allowed for bilevel images in the IOCA Function Set specified for the image can be used. If the datastream does not specify the function set, any architecturally valid Image Encoding parameter values can be used, except Solid Fill Rectangle. The Solid Fill Rectangle algorithm is not needed, since omitting the transparency mask achieves the same effect as setting all the transparency mask image points to 1. Completely removing the image achieves the same effect as Setting all transparency mask image points to 0. If the Image Encoding parameter is missing, the default encoding (no compression and RIDIC)

| applies.
|

Begin Transparency Mask

The Begin Transparency Mask defines the beginning of the transparency mask.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'8E'	Begin Transparency Mask	M
1	UBIN	LENGTH	X'00'	Length of the parameters to follow	M

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-8E0F **Invalid sequence**

Condition: A Begin Transparency Mask has appeared out of sequence or more than once within a tile or a non-tiled image.

End Transparency Mask

The End Transparency Mask defines the end of a transparency mask.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'8F'	End Transparency Mask	M
1	UBIN	LENGTH	X'00'	Length of the parameters to follow	M

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-8F0F **Invalid sequence**

Condition: An End Transparency Mask is missing after a Begin Transparency Mask has been encountered, or it appeared out of sequence.

Image data elements

| This section describes the parameters used to
| carry the image data elements.

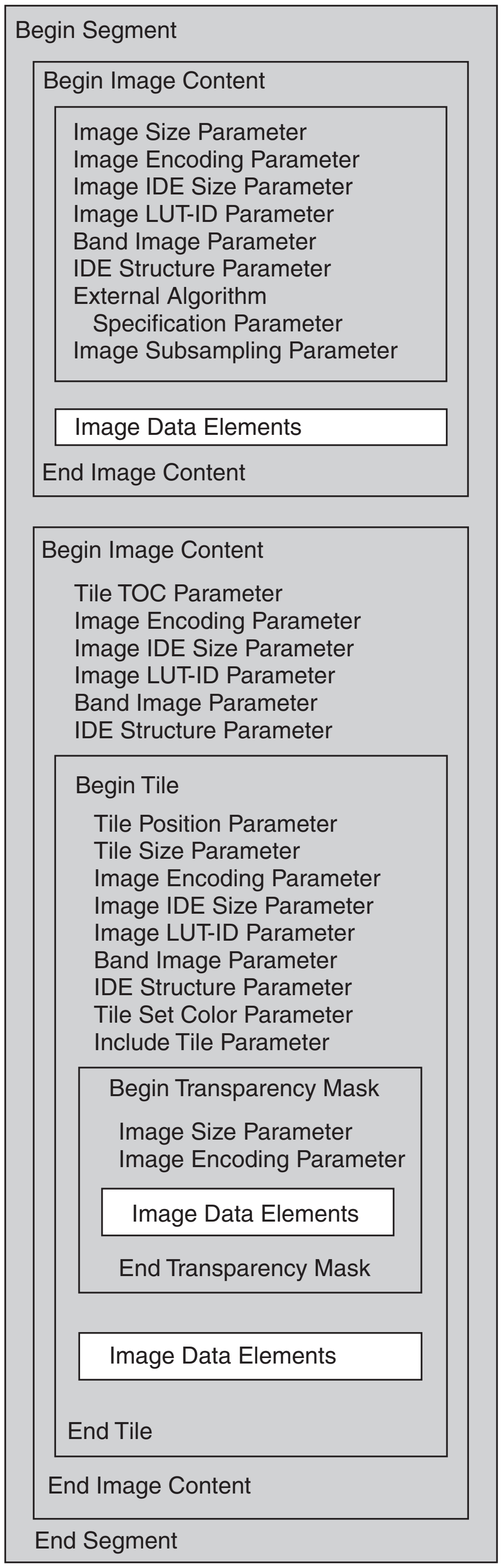


Image Data

| The Image Data self-defining field is an element of the image content, and is a set
| of IDEs. Multiple Image Data self-defining fields of the same type can be contained
| in a single untiled image content, a single tile, or a single transparency mask.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0–1	CODE	ID	X'FE92'	Image Data	M
2–3	UBIN	LENGTH	X'0001' – X'FFFF'	Length of the image data to follow	M
4–*n*		DATA	Any	Image data elements. The data in this self-defining field is recorded, compressed and ordered as specified by the Image Encoding parameter. For some function sets, the data can also be subsampled as described by the Image Subsampling parameter.	M

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-9201 **Invalid existence of Image Data**

| **Condition:** Image Data should not be present, as in the case when the image data is in bands.

EC-920F **Invalid sequence:**

Condition: Image Data is missing, or it appeared out of sequence.

Band Image Data

The Band Image Data self-defining field is an element of the image content. It is a set of IDEs, typically having similar attributes to each other.

Band Image Data must appear within an image content or a tile for each band defined by the Band Image parameter. The bands must appear in the sequential order of their band numbers. The Band Image parameter must exist in the image content if this parameter is used. See "Band Image" on page 40 for more detail.

If the data for a particular band exceeds the length of a single Band Image Data, the remaining data is contained in one or more Band Image Data parameters having the same band number, and following in sequence.

Syntax

Offset	Type	Name	Range	Meaning	M/O
0–1	CODE	ID	X'FE9C'	Band Image Data	M
2–3	UBIN	LENGTH	X'0003' – X'FFFF'	Length of the parameters to follow	M
4	CODE	BANDNUM	X'01' – X'FD'	Band number	M
5–6			X'0000'	Reserved; should be zero	M
7–*n*		DATA	Any	Image data elements for the band. The data in this self-defining field is recorded, compressed and ordered as specified by the Image Encoding parameter. For some function sets, the data can also be subsampled as described by the Image Subsampling parameter.	O

Note: If the Band Image Data contains no data (the length is X'0003') for a certain band, all the uncompressed image data elements in the band are zero. For such a band, only a single Band Image Data parameter can appear; otherwise exception EC-9C0F occurs.

Exception conditions

The following exception conditions cause the standard action to be taken:

EC-0003 **Invalid length**

Condition: The LENGTH value is not in the valid range.

EC-0004 **Invalid parameter value**

Condition: The BANDNUM value is not in the valid range.

EC-9C01 **Invalid existence of Band Image Data**

Condition: Band Image Data should not be present, as in the case when the image data is not in bands.

EC-9C0F **Invalid sequence**

Condition: Band Image Data is missing, or it appeared out of sequence.

EC-9C17 **Invalid number/sequence of Band Image Data**

Condition: The band numbers of a band image do not appear for the number of bands defined in the Band Image parameter, or do not appear in succeeding order.

Chapter 6. Exception conditions and actions

This chapter outlines the exception conditions and exception actions for IOCA, and summarizes:
- Exception conditions causing the common standard action
- Exception conditions causing unique standard actions

An exception condition is either mandatory or optional. If a function is supported and a mandatory exception condition for the function is detected, the process must notify the controlling environment. If an optional exception condition for the function is detected, the process may or may not need to notify the controlling environment.

The table in "Mandatory or optional exception conditions" on page 88 summarizes for each IOCA exception condition, whether it is mandatory or optional. Also shown in the table is whether the two primary IOCA controlling environments—MO:DCA and IPDS—would consider the exception condition to be mandatory or optional.

Exception conditions

Exception conditions include violations of the following:
- Syntax
- Parameter value
- Self-defining field sequence

Exceptions are represented in the following format:

eee-xxxx

where:

eee identifies the exception group. The exception group can be one of the following:

EC image segment exception

xxxx is the exception code (two-byte hexadecimal value).

There are two types of exception conditions:
- Those that cause the common standard action
- Those that cause unique standard actions

The exception conditions are summarized in the following sections. Unique exception conditions and actions that are related to a specific element are described in the corresponding section for the element.

image segment exception conditions

This exception occurs when a violation of format, parameter, or sequence of execution to this architecture is detected in the image segment.

The exception is represented in the following format:

: EC-*xxxx*

where:

EC	identifies an image segment exception condition.
xxxx	is the image segment exception condition code (two-byte hexadecimal value).

The following image segment exception conditions are defined:
- Exception conditions that cause the common standard action
- Exception conditions that cause unique standard actions

The IOCA process model recovers the exception condition according to the severity of the exception. The severity of an exception depends on the image data parameter or the image data.

If an exception action is defined in the corresponding section, the action is taken first, and the exception condition is kept until it is reported to the controlling environment.

If the action is not defined in the corresponding section, the rest of the image segment is not processed and the exception condition is reported immediately to the controlling environment.

Exception actions

The IOCA process model responds with an exception action when it detects an exception condition.

An exception condition prompts either of the following kinds of actions:
- An exception action defined by IOCA
- An alternative action that is defined outside the IOCA process model

The controlling environment governs which way the IOCA process model responds to the exception conditions. For example, the IPDS architecture has a command to specify whether the IPDS-defined page continuation action or the IOCA-defined exception action is to be taken.

IOCA process model actions

The IOCA process model recovers the exception condition according to the severity of the exception. The severity depends on the condition where the exception is detected.

The exception conditions are reset after the controlling environment has been notified.

Unique standard actions

If a unique exception action is defined for the exception condition (such as for EC-9401 and EC-9511), the IOCA process model:
- Notifies the controlling environment of the condition
- Performs the defined unique action

Common standard action

If no unique exception action is defined, the IOCA process model:
- Notifies the controlling environment of the condition
- Does not process the rest of the image segment

The exception conditions are reset after the controlling environment has been notified.

Exception conditions causing the common standard action

EC-0001 Invalid or unsupported code within an Image Segment

Explanation: An invalid or unsupported self-defining field is detected within the Image Segment.

EC-0002 Reserved bits or bytes are not zeros

Explanation: Reserved bits or bytes are not zeros in the Image Data parameter within the Image Segment.

Note: This exception condition is optional.

EC-0003 The LENGTH value is not in the valid range

Explanation: The LENGTH value for the Begin/End Segment, Begin/End Image Content, image data parameter, image data, band image data, or Numbered Image Data is not in the valid range.

EC-0004 Invalid parameter value

Explanation: A parameter value of an Image Data parameter is not in the valid range.

EC-0005 Invalid Length

Explanation: The LENGTH value is not in the valid, function-set specified range. EC-0005 is optional: IOCA receivers can generate EC-0003 instead of EC-0005.

EC-xx0F Invalid sequence

Explanation: The sequence of self-defining fields is not correct within an image segment. This exception condition is also raised when a mandatory or necessary self-defining field is missing, or when a self-defining field other than image data, band image data, or Numbered Image Data appears more than once in an image segment. Value *xx* in the exception code depends on the image data parameter in which the exception occurred. The parameter code is placed in this *xx* position.

For example:
- **EC-700F** for the Begin Segment (see page “Exception conditions” on page 23)
- **EC-710F** for the End Segment (see Note below and page“Exception conditions” on page 24)
- **EC-8C0F** for the Begin Tile (see page “Exception conditions” on page 57)
- **EC-8D0F** for the End Tile (see page “Exception conditions” on page 58)
- **EC-8E0F** for the Begin Transparency Mask (see page “Exception conditions” on page 76)
- **EC-8F0F** for the End Transparency Mask (see page “Exception conditions” on page 77)
- **EC-910F** for the Begin Image Content (see page “Exception conditions” on page 26)
- **EC-920F** for the Image Data (see Note below and page “Exception conditions” on page 79)
- **EC-930F** for the End Image Content (see page “Exception conditions” on page 28)
- **EC-940F** for the Image Size parameter (see page “Exception conditions” on page 31)
- **EC-950F** for the Image Encoding parameter (see page “Exception conditions” on page 36)
- **EC-960F** for the IDE Size parameter (see page “Exception conditions” on page 38)
- **EC-970F** for the Image LUT-ID parameter (see page “Exception conditions” on page 39)
- **EC-980F** for the Band Image parameter (see page “Exception conditions” on page 41)
- **EC-9B0F** for the IDE Structure parameter (see page “Exception conditions” on page 45)
- **EC-9C0F** for the Band Image Data (see Note below and page “Exception conditions” on page 80)
- **EC-9F0F** for the External Algorithm Specification parameter (see page “Exception conditions” on page 47)
- **EC-B50F** for the Tile Position (see page “Exception conditions” on page 59)
- **EC-B60F** for the Tile Size (see page “Exception conditions” on page 62)
- **EC-B70F** for the Tile Set Color (see page “Exception conditions” on page 68)
- **EC-B80F** for the Include Tile (see page “Exception conditions” on page 69)
- **EC-BB0F** for the Tile TOC (see page “Exception conditions” on page 72)
- **EC-CE0F** for the Image Subsampling parameter (see page “Exception conditions” on page 52)

Note: This exception condition is not raised when the indicated self-defining field appears more than once.

EC-xx10 Invalid or unsupported Image Data parameter value

Explanation: The specified value for an image data parameter is not valid in the architecture or function sets, or is not supported by the implementation. Value *xx* in the exception code depends on the image data parameter in which the exception occurred. The parameter code is placed in this *xx* position.

For example:
- **EC-9410** for the Image Size parameter (see page "Exception conditions" on page 31)
- **EC-9510** for the Image Encoding parameter (see page "Exception conditions" on page 36)
- **EC-9610** for the IDE Size parameter (see page "Exception conditions" on page 38)
- **EC-9710** for the Image LUT-ID parameter (see page "Exception conditions" on page 39)
- **EC-9810** for the Band Image parameter (see page "Exception conditions" on page 41)
- **EC-9B10** for the IDE Structure parameter (see page "Exception conditions" on page 45)
- **EC-9F10** for the External Algorithm Specification parameter (see page "Exception conditions" on page 47)
- **EC-B510** for the Tile Position (see page "Exception conditions" on page 59)
- **EC-B610** for the Tile Size (see page "Exception conditions" on page 62)
- **EC-B710** for the Tile Set Color (see page "Exception conditions" on page 68)
- **EC-BB10** for the Tile TOC (see page "Exception conditions" on page 72)
- **EC-CE10** for the Image Subsampling parameter (see page "Exception conditions" on page 52)

EC-xx11 Inconsistent Image Data parameters, or inconsistent Image Data parameter and Image Data

Explanation: The specified value for an image data parameter is not consistent with a value specified in another image data parameter or with the image data following it. Value *xx* in the exception code depends on the image data parameter in which the exception occurred. The parameter code is placed in this *xx* position.

For example:
- **EC-9411** for the Image Size parameter (see page "Exception conditions" on page 31)
- **EC-9611** for the IDE Size parameter (see page "Exception conditions" on page 38)
- **EC-9F11** for the External Algorithm Specification parameter (see page "Exception conditions" on page 47)
- **EC-B511** for the Tile Position (see page "Exception conditions" on page 59)
- **EC-B611** for the Tile Size (see page "Exception conditions" on page 62)
- **EC-B711** for the Tile Set Color (see page "Exception conditions" on page 68)
- **EC-B811** for the Include Tile (see page "Exception conditions" on page 69)
- **EC-BB11** for the Tile TOC (see page "Exception conditions" on page 72)

EC-9201 Invalid existence of Image Data

Explanation: Image Data should not be present, as in the case when the image data is in bands.

EC-9801 Invalid Band Image parameter and Image Subsampling parameter coexistence

Explanation: In some function sets, Band Image parameter and Image Subsampling parameter cannot coexist in the same Image Content.

EC-9814 Invalid number of bands and bit counts

Explanation: The number of BITCNT parameters is not equal to the BCOUNT in the Band Image parameter.

EC-9815 Invalid IDE size

Explanation: The IDE size, determined by the Band Image parameter, does not match the IDE Size parameter.

EC-9B18 Invalid IDE Structure parameter

Explanation: The IDE size in the IDE Structure parameter, determined by the sum of SIZE1 through SIZE3, does not match that of the IDE Size parameter.

One of the following conditions occurs:
- The sum of SIZE1 through SIZE4 does not match the IDE size specified by the IDE Size parameter.
- Color space is CMYK and SIZE4 is missing.

| • SIZE4 is present and the color space is not CMYK.

EC-9C01 Invalid existence of Band Image Data

Explanation: Band Image Data should not be present, as in the case when the Image Data is not in bands.

EC-9C17 Invalid number/sequence of Band Image Data

Explanation: The band numbers of a band image do not appear for the number of bands defined in the Band Image parameter, or do not appear in succeeding order.

EC-9F01 Missing External Algorithm Specification parameter or Image Encoding parameter

Explanation: An External Algorithm Specification parameter exists without a corresponding Image Encoding parameter, or an Image Encoding parameter exists that requires an External Algorithm Specification parameter, which cannot be found.

EC-CE01 Invalid Band Image parameter and Image Subsampling parameter coexistence

Explanation: In some function sets, Band Image parameter and Image Subsampling parameter cannot coexist in the same Image Content.

Exception conditions causing unique standard actions

EC-9401 Inconsistent Image Size parameter value and Image Data

Explanation: The size detected in the image data is different from the HSIZE or VSIZE value of the Image Size parameter.

System action: The size detected from the image data is used.

EC-9511 Inconsistent Image Data parameters, or inconsistent Image Data parameter and Image Data

Explanation: The decoder encountered one of the following conditions when decompressing the image data:

- The image data is not encoded according to the compression or recording algorithm specified in the Image Encoding parameter.
- The image data cannot be decoded successfully using the size values specified in the Image Size parameter. This condition applies to compression or recording algorithms which do not permit the image size to be encoded in the image data.
- The image data is not in complete accordance with the compression algorithm specified in the Image Encoding parameter.
- | Image is encoded using the algorithm specified in the Image Encoding Parameter, but uses a function of the algorithm that is unsupported by the receiver.

System action: Receivers should attempt to present or make use of all successfully decompressed image data. Note that the resulting partial image might differ from the original image.

EC-A902 Write outside of the Image Presentation Space

Explanation: A portion of the extracted image is written outside the Image Presentation Space.

System action: The portion inside the Image Presentation Space is presented, and the rest is discarded.

Mandatory or optional exception conditions

Exception	**IOCA**	**MO:DCA**	**IPDS**
EC-0001	Mandatory	Same as IOCA	Same as IOCA
EC-0002	Optional	Same as IOCA	Same as IOCA
EC-0003	Mandatory	Same as IOCA	Same as IOCA
EC-0004	Mandatory	Same as IOCA	Same as IOCA
EC-0005	Optional*	Same as IOCA	Same as IOCA
EC-*xx*0F	Mandatory	Same as IOCA	Same as IOCA
EC-*xx*10	Mandatory	Same as IOCA	Same as IOCA
EC-*xx*11	Mandatory	Same as IOCA	Same as IOCA
EC-9201	Mandatory	Same as IOCA	Same as IOCA
EC-9401	Mandatory	Same as IOCA	Same as IOCA
EC-9801	Mandatory	Same as IOCA	Same as IOCA
EC-9814	Mandatory	Same as IOCA	Same as IOCA
EC-9815	Mandatory	Same as IOCA	Same as IOCA
EC-9A15	Mandatory	Same as IOCA	Same as IOCA
EC-9B18	Mandatory	Same as IOCA	Same as IOCA
EC-9C01	Mandatory	Same as IOCA	Same as IOCA
EC-9C17	Mandatory	Same as IOCA	Same as IOCA

Exception	IOCA	MO:DCA	IPDS
EC-9F01	Mandatory	Same as IOCA	Same as IOCA
EC-A902	Mandatory	Same as IOCA	Same as IOCA
EC-CE01	Mandatory	Same as IOCA	Same as IOCA
*Receiver can generate EC-0003 instead of EC-0005.			

Mandatory or optional exception conditions

Chapter 7. Compliance

This chapter:
- Describes the concept of IOCA function sets
- Lists the product compliance rules
- Defines IOCA Function Sets FS10, FS11, FS20, FS42 and FS45

Function sets

A *function set* is a set of self-defining fields that describes an image object. Specifically, it is a definition of the image segment: which parameters the image segment should consist of, and what values each parameter should have. The image object described in the function set can thus be processed in different controlling environments.

Each function set has an identification. With that identification, products determine the level of support they must provide to generate or receive IOCA image objects.

Table 4. Function set Identification

ID	**Description**	**Function sets currently defined**
0*x*	Stand-alone	
1*x*	Carried by presentation-level data stream, non-tiled	FS10, FS11
2*x*	Library/resource	FS20
3*x*	Reserved	
4*x*	Carried by presentation-level data stream, tiled	FS42, FS45
Note: *x* is assigned in ascending numerical order from zero.		

IOCA defines six function sets: FS10, FS11, FS20, FS40, FS42 and FS45.
- Function Set 10 is intended for bilevel images.
- Function Sets 11 and 20 cover bilevel, grayscale, and color images.
- Function Set 40 covers tiled bilevel images.
- Function Set 42 covers tiled bilevel images and tiled CMYK images with one bit per spot (SIZE1=SIZE2=SIZE3=SIZE4=1, IDESZ=4).
- Function set 45 carries tiled bilevel and CMYK images. CMYK images can be either one or eight bits per spot (IDESZ=4 or IDESZ=32).

Function Set 11 is a superset of Function Set 10. Function Set 45 is a superset of Function Set 42. Function Set 42 is a superset of Function Set 40. There are no other relationships among the function sets.

Products that conform to an IOCA function set must meet the following criteria:
- Products that generate IOCA image objects can only use the IOCA self-defining fields and parameter values that are defined in the corresponding function set definition
- Products that receive IOCA image objects must be capable of accepting any IOCA image object that conforms to the corresponding function set definition.

Function sets

The following function sets are part of SAA®:
- Function Set 10
- Function Set 11
- Function Set 20

The following sections show the self-defining fields that each function set contains and the acceptable values for each field.

IOCA Function Set 10 (IOCA FS10)

Function Set 10 describes bilevel images. This function set is carried by the MO:DCA-P and IPDS controlling environments. The permissible parameter groupings in FS10 are defined as follows:

Table 5. Function Set 10 structure

		X'70'	Begin Segment parameter		
		X'91'	Begin Image Content parameter		
+		X'94'	Image Size parameter		
+	[	X'95'	Image Encoding parameter		]
+	[	X'96'	IDE Size parameter		]
+	[	X'97'	Image LUT-ID parameter		]
		X'FE92'	Image Data	(S)	
		X'93'	End Image Content parameter		
		X'71'	End Segment parameter		

The self-defining fields and values acceptable for FS10 are shown in the following table.

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Begin Segment	ID (1)	X'70'	
	LENGTH (1)	X'00'	
Begin Image Content	ID (1)	X'91'	
	LENGTH (1)	X'01'	
	OBJTYPE (1)	X'FF'	IOCA
Image Size parameter	ID (1)	X'94'	
	LENGTH (1)	X'09'	
	UNITBASE (1)	X'00' – X'02'	
	HRESOL (2)	X'0000' – X'FFFF'	
	VRESOL (2)	X'0000' – X'FFFF'	
	HSIZE (2)	X'0000' – X'FFFF'	
	VSIZE (2)	X'0000' – X'FFFF'	
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02'	
	COMPRID (1)	X'01', X'03', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read **X'03'** No compression **X'82'** G4 MMR—Modified Modified Read
	RECID (1)	X'01'	RIDIC
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01'	1bit/IDE

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image LUT-ID parameter	ID (1)	X'97'	
	LENGTH (1)	X'01'	
	LUTID (1)	X'00'	Standard LUT-ID
Image Data	ID (2)	X'FE92'	
	LENGTH (2)	X'0001' – X'FFFF'	
	DATA	Any	IDEs (see Note)
End Image Content	ID (1)	X'93'	
	LENGTH (1)	X'00'	
End Segment	ID (1)	X'71'	
	LENGTH (1)	X'00'	
Note: IDE value 0 represents an *insignificant* image point, and 1 represents a *significant* image point. The controlling environment determines how to interpret each value.			

IOCA Function Set 11 (IOCA FS11)

Function Set 11 is a superset of Function Set 10, and describes bilevel, grayscale, and color images. This function set is carried by the MO:DCA IS/2 controlling environment. The permissible parameter groupings in FS11 are defined as follows:

Table 6. Function Set 11 structure

		X'70'	Begin Segment parameter	
		X'91'	Begin Image Content parameter	
+		X'94'	Image Size parameter	
+	[	X'95'	Image Encoding parameter	]
+	[	X'96'	IDE Size parameter	]
+	[	X'97'	Image LUT-ID parameter	]
+	[	X'98'	Band Image parameter	]
+	[	X'9B'	IDE Structure parameter	]
+	[	X'9F'	External Algorithm Specification parameter	]
+	[	X'FECE'	Image Subsampling parameter	]
			Image Data or Band Image Data	(S)
		X'93'	End Image Content parameter	
		X'71'	End Segment parameter	

The self-defining fields and values acceptable for FS11 are shown in the following table.

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Initial parameters:			
Begin Segment	ID (1)	X'70'	
	LENGTH (1)	X'00'	
Begin Image Content	ID (1)	X'91'	
	LENGTH (1)	X'01'	
	OBJTYPE (1)	X'FF'	IOCA
Image Size parameter	ID (1)	X'94'	
	LENGTH (1)	X'09'	
	UNITBASE (1)	X'00' – X'02'	
	HRESOL (2)	X'0000' – X'7FFF'	
	VRESOL (2)	X'0000' – X'7FFF'	
	HSIZE (2)	X'0000' – X'7FFF'	
	VSIZE (2)	X'0000' – X'7FFF'	

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'03'	
	COMPRID (1)	X'02', X'03', X'08', X'0A', X'82', X'83'	**X'01'** InfoPrint MMR—Modified Modified Read (see Note 1) **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) (see Note 1) **X'0A'** Concatenated ABIC (see Note 2) **X'82'** G4 MMR—Modified Modified Read (see Note 1) **X'83'** JPEG algorithms (see Note 3)
	RECID (1)	X'01'	RIDIC
	BITORDR	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01', X'04', X'08', X'18'	**X'01'** 1 bit/IDE **X'04'** 4 bits/IDE **X'08'** 8 bits/IDE **X'18'** 24 bits/IDE

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
External Algorithm Specification parameter	ID (1)	X'9F'	
	LENGTH (1)	X'0A'	
	ALGTYPE (1)	X'10'	Compression algorithm specification
	RESERVED (1)	X'00'	Should be zero
	COMPRID (1)	X'83'	JPEG algorithms
	RESERVED (3)	X'000000'	Should be zero
	MARKER (1)	X'C0' – X'C2', X'C9' – X'CA'	Non-differential Huffman coding: **X'C0'** Baseline DCT **X'C1'** Extended sequential DCT **X'C2'** Progressive DCT Non-differential arithmetic coding: **X'C9'** Extended sequential DCT **X'CA'** Progressive DCT
	RESERVED (3)	X'000000'	Should be zero
Notes on the initial parameters: 1. ABIC (Bilevel Q-Coder), InfoPrint MMR—Modified Modified Read, and G4 MMR—Modified Modified READ are applicable only to images whose IDE size is 1 bit/IDE, otherwise exception condition EC-9611 is raised. 2. Concatenated ABIC is applicable only to images whose IDE size is 4 or 8 bits/IDE, otherwise exception condition EC-9611 is raised. 3. The JPEG baseline DCT-based algorithm is applicable only to images whose IDE size is 8 bits/IDE, while the other DCT-based algorithms are applicable only to images whose IDE size is 8 or 24 bits/IDE; otherwise exception condition EC-9611 is raised.			
Parameters used when IDESZ=1:			
Image LUT-ID parameter	ID (1)	X'97'	
	LENGTH (1)	X'01'	
	LUTID (1)	X'00'	Standard LUT-ID
Band Image parameter (see General Note at the end of the table)	ID (1)	X'98'	
	LENGTH (1)	X'02'	
	BCOUNT (1)	X'01'	One band
	BITCNT (1)	X'01'	1 bit/IDE

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
IDE Structure parameter	ID (1)	X'9B'	
	LENGTH (1)	X'06' – X'08'	
	FLAGS (1)		
	ASFLAG	B'0'	Additive
	GRAYCODE	B'0'	No gray coding
	RESERVED	B'000000'	Should be zero
	FORMAT (1)	X'02', X'12'	**X'02'** YCrCb **X'12'** YCbCr
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'01'	1 bit/IDE
	SIZE2 (1)	X'00'	0 bit/IDE
	SIZE3 (1)	X'00'	0 bit/IDE
Image Subsampling parameter (see General note at the end of the table)	ID (2)	X'FECE'	
	LENGTH (2)	X'0000', X'0004'	
	ID (1)	X'01'	Sampling ratios
	LENGTH (1)	X'02'	
	HSAMPLE (1)	X'01'	
	VSAMPLE (1)	X'01'	
Parameters used when IDESZ=4 or IDESZ=8			
Band Image parameter (see General note at the end of the table)	ID (1)	X'98'	
	LENGTH (1)	X'02'	
	BCOUNT (1)	X'01'	One band
	BITCNT (1)	X'04', X'08'	**X'04'** 4 bits/IDE **X'08'** 8 bits/IDE
IDE Structure parameter	ID (1)	X'9B'	
	LENGTH (1)	X'06' – X'08'	
	FLAGS (1)		
	ASFLAG	B'0'	Additive
	GRAYCODE	B'0' – B'1'	**B'0'** No gray coding **B'1'** Gray coding (see Note 1)
	RESERVED	B'000000'	Should be zero
	FORMAT (1)	X'08'	**X'02'** YCrCb (see Note 2) **X'12'** YCbCr (see Note 2)
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'04', X'08'	**X'04'** 4 bits/IDE **X'08'** 8 bits/IDE
	SIZE2 (1)	X'00'	0 bit/IDE
	SIZE3 (1)	X'00'	0 bit/IDE

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image Subsampling parameter (see General note at the end of the table)	ID (2)	X'FECE'	
	LENGTH (2)	X'0000', X'0004'	
	ID (1)	X'01'	Sampling ratios
	LENGTH (1)	X'02'	
	HSAMPLE (1)	X'01'	
	VSAMPLE (1)	X'01'	
Notes on parameters used when IDESZ=4 or IDESZ=8: 1. Gray coding is valid only for the Concatenated ABIC algorithm, otherwise exception condition EC-9B10 is raised. 2. Grayscale images only. Grayscale IDEs are composed of the Y component only of the YCrCb or YCbCr color model.			
Parameters used when IDESZ=24:			
Band Image parameter (see General note at the end of the table)	ID (1)	X'98'	
	LENGTH (1)	X'02'	
	BCOUNT (1)	X'01'	One band
	BITCNT (1)	X'18'	24 bits/IDE
or:			
Band Image parameter (see General note at the end of the table)	ID (1)	X'98'	
	LENGTH (1)	X'04'	
	BCOUNT (1)	X'03'	3 bands: R,G,B or Y,Cr,Cb or Y,Cb,Cr
	BITCNT (1)	X'08'	8 bits/IDE for R or Y band
	BITCNT (1)	X'08'	8 bits/IDE for G or Cr or Cb band
	BITCNT (1)	X'08'	8 bits/IDE for B or Cb or Cr band

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
IDE Structure parameter	ID (1)	X'9B'	
	LENGTH (1)	X'08'	
	FLAGS (1)		
	ASFLAG	B'0'	Additive
	GRAYCODE	B'0'	No gray coding
	RESERVED	B'000000'	Should be zero
	FORMAT (1)	X'01' – X'02', X'12'	**X'01'** RGB **X'02'** YCrCb **X'12'** YCbCr
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'08'	8 bits of the IDE for the R or Y component
	SIZE2 (1)	X'08'	8 bits of the IDE for the G or Cr or Cb component
	SIZE3 (1)	X'08'	8 bits of the IDE for the B or Cb or Cr component
Image Subsampling parameter (see General note at the end of the table)	ID (2)	X'FECE'	
	LENGTH (2)	X'0000', X'0004', X'0006', X'0008'	
	ID (1)	X'01'	Sampling ratios
	LENGTH (1)	X'02', X'04', X'06'	
	HSAMPLE1 (1)	X'01' – X'02'	**X'02'** YCrCb and YCbCr color models only
	VSAMPLE1 (1)	X'01'	
	HSAMPLE2 (1)	X'01'	
	VSAMPLE2 (1)	X'01'	
	HSAMPLE3 (1)	X'01'	
	VSAMPLE3 (1)	X'01'	
Final parameters:			
image data	ID (2)	X'FE92'	
	LENGTH (2)	X'0001' – X'FFFF'	
	DATA	Any	IDEs (see Note on the final parameters)
or:			
band image data (BCOUNT=1 only)	ID (2)	X'FE9C'	
	LENGTH (2)	X'0004' – X'FFFF'	
	BANDNUM (1)	X'01'	One band
	RESERVED (2)	X'0000'	Should be zero
	DATA	Any	IDEs (see Note on the final parameters)
or:			

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
band image data (BCOUNT=3 only)	ID (2)	X'FE9C'	
	LENGTH (2)	X'0004' – X'FFFF'	
	BANDNUM (1)	X'01'	Band containing the R or Y component of the IDEs
	RESERVED (2)	X'0000'	Should be zero
	DATA	Any	R or Y component of the IDEs
	ID (2)	X'FE9C'	
	LENGTH (2)	X'0004' – X'FFFF'	
	BANDNUM (1)	X'02'	Band containing the G or Cr or Cb component of the IDEs
	RESERVED (2)	X'0000'	Should be zero
	DATA	Any	G or Cr or Cb component of the IDEs
	ID (2)	X'FE9C'	
	LENGTH (2)	X'0004' – X'FFFF'	
	BANDNUM (1)	X'03'	Band containing the B or Cb or Cr component of the IDEs
	RESERVED (2)	X'0000'	Should be zero
	DATA	Any	B or Cb or Cr component of the IDEs
End Image Content	ID (1)	X'93'	
	LENGTH (1)	X'00'	
End Segment	ID (1)	X'71'	
	LENGTH (1)	X'00'	
Note on the final parameters: With IDESZ=1 and LUTID=0, IDE value 0 represents an *insignificant* image point, and 1 represents a *significant* image point. The controlling environment determines how to interpret each value.			
General note: In this function set, the Image Subsampling parameter and the Band Image parameter cannot coexist within the same image content; otherwise exception condition EC-9801 or EC-CE01 is raised.			

IOCA Function Set 20 (IOCA FS20)

Function Set 20 describes bilevel, grayscale, and color images. This function set is carried by the MO:DCA-L controlling environment. The permissible parameter groupings in FS20 are defined as follows:

Table 7. Function Set 20 structure

		X'70'	Begin Segment parameter		
		X'91'	Begin Image Content parameter		
+		X'94'	Image Size parameter		
+	[	X'95'	Image Encoding parameter		]
+	[	X'96'	IDE Size parameter		]
+	[	X'97'	Image LUT-ID parameter		]
+	[	X'9B'	IDE Structure parameter		]
		X'FE92'	Image Data	(S)	
		X'93'	End Image Content parameter		
		X'71'	End Segment parameter		

Its acceptable self-defining fields and parameter values are shown in the following table.

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Initial parameters:			
Begin Segment	ID (1)	X'70'	
	LENGTH (1)	X'00'	
Begin Image Content	ID (1)	X'91'	
	LENGTH (1)	X'01'	
	OBJTYPE (1)	X'FF'	IOCA
Image Size parameter	ID (1)	X'94'	
	LENGTH (1)	X'09'	
	UNITBASE (1)	X'00' – X'FF02'	
	HRESOL (2)	X'0000' – X'FFFF'	
	VRESOL (2)	X'0000' – X'FFFF'	
	HSIZE (2)	X'0000' – X'FFFF'	
	VSIZE (2)	X'0000' – X'FFFF'	

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02'	
	COMPRID (1)	X'01', X'03', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read (see Note 1) **X'03'** No Compression **X'82'** G4 MMR—Modified Modified Read (see Note 1)
	RECID (1)	X'01', X'03'	**X'01'** RIDIC **X'03'** Bottom-to-Top (see Note 2)
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01', X'04', X'08', X'18'	**X'01'** 1 bit/IDE **X'04'** 4 bits/IDE **X'08'** 8 bits/IDE **X'18'** 24 bits/IDE
Notes on the initial parameters: 1. InfoPrint MMR—Modified Modified Read and G4 MMR—Modified Modified READ are applicable only to images whose IDE size is 1 bit/IDE; otherwise exception condition EC-9611 is raised. 2. Bottom-to-Top is used only in conjunction with No Compression; otherwise exception condition EC-9510 is raised.			
Parameters used when IDESZ=1:			
Image LUT-ID parameter	ID (1)	X'97'	
	LENGTH (1)	X'01'	
	LUTID (1)	X'00' – X'01'	
Parameters used when IDESZ=4 or IDESZ=8:			
Image LUT-ID parameter	ID (1)	X'97'	
	LENGTH (1)	X'01'	
	LUTID (1)	X'01'	
Parameters used when IDESZ=24:			
Image LUT-ID parameter	ID (1)	X'97'	
	LENGTH (1)	X'01'	
	LUTID (1)	X'00'	Standard Image LUT-ID

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
IDE Structure parameter	ID (1)	X'9B'	
	LENGTH (1)	X'08'	
	FLAGS (1)	X'00'	Additive and No gray coding
	FORMAT (1)	X'01'	RGB
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'08'	8 bits of the IDE for the R component
	SIZE2 (1)	X'08'	8 bits of the IDE for the G component
	SIZE3 (1)	X'08'	8 bits of the IDE for the B component
Final parameters:			
image data	ID (2)	X'FE92'	
	LENGTH (2)	X'0001' – X'FFFF'	
	DATA	Any	IDEs (see Note)
End Image Content	ID (1)	X'93'	
	LENGTH (1)	X'00'	
End Segment	ID (1)	X'71'	
	LENGTH (1)	X'00'	
Note on the final parameters: With IDESZ=1 and LUTID=0, IDE value 0 represents an *insignificant* image point, and 1 represents a *significant* image point. The controlling environment determines how to interpret each value.			

IOCA Function Set 40 (IOCA FS40)

Function Set 40 is a subset of Function Set 45. It describes tiled images with one bit per spot (color space YCbCr or YCrCb, IDESZ=1). This function set is carried by the MO:DCA-P controlling environment. The permissible parameter groupings in FS40 are defined as follows:

Table 8. Function Set 40 structure

	X'70'	Begin Segment parameter		
	X'91'	Begin Image Content parameter		
	X'FEBB'	Tile TOC parameter		
[	X'95'	Image Encoding parameter		]
[	X'96'	IDE Size parameter		]
[	X'9B'	IDE Structure parameter		]
[		Tiles	(S)	]
	X'93'	End Image Content parameter		
	X'71'	End Segment parameter		

Table 9. Tile structure

	X'8C'	Begin Tile parameter		
	X'B5'	Tile Position parameter		
	X'B6'	Tile Size parameter		
[	X'95'	Image Encoding parameter		]
[	X'96'	IDE Size parameter		]
[	X'9B'	IDE Structure parameter		]
[	X'FE92'	Image Data	(S)	]
	X'8D'	End Tile		

Notes:

1. Note that the parameters in the above diagram must come in the specified order. Even though the general IOCA architecture allows different ordering for some of the parameters, the FS40 specification is more restrictive. If the parameters are given in a different order, an out-of-sequence exception is raised.
2. In the context of FS40, Image Size parameter, Image Subsampling parameter, External Algorithm parameter, and Image LUT ID parameter cause EC-0001 exception (Invalid parameter) to occur. If the first parameter after the Begin Image Content parameter is not the Tile TOC parameter, the image is not a tiled image and any of the tile-specific parameters (Tile TOC parameter, Begin Tile parameter, etc.) cause EC-0001 to occur.
3. Image Encoding parameter, IDE Size parameter, Band Image parameter, and IDE Structure parameter are shown as optional and can possibly be specified in two places. The specification within a tile takes precedence over a specification outside of the tile.
4. If IDE Size parameter is not present, the default IDE size is one bit per pel (bilevel image).
5. If the Image Encoding parameter is not present, the default compression algorithm is X'03' (No Compression). The recording algorithm defaults to X'01' (RIDIC); and the bit and byte orders default to zero.

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Initial parameters in Function Set 40:			
Begin Segment	ID (1)	X'70'	
	LENGTH (1)	X'00'	
Begin Image Content	ID (1)	X'91'	
	LENGTH (1)	X'01'	
	OBJTYPE (1)	X'FF'	IOCA
Tile TOC parameter	ID (2)	X'FEBB'	
	LENGTH (2)		
	Reserved (2)	X'0000'	Reserved; should be set to zero
	Either zero repeating groups, or one per tile in the following format:		
	XOFFSET (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile origin
	YOFFSET (4)	X'00000000' – X'7FFFFFFF'	Vertical tile origin
	THSIZE (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile size
	TVSIZE (4)	X'00000000' – X'7FFFFFFF'	Vertical tile size
	RELRES (1)	X'01'	Relative resolution
	COMPR (1)		Compression algorithm
	DATAPOS (8)		File offset to the beginning of the tile

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'03'	
	COMPRID (1)	X'01', X'03', X'08', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read (see General note) **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) (see General note) **X'82'** G4 MMR—Modified Modified Read (see General note)
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01'	**X'01'** 1 bit/IDE
Initial parameters in a tile:			
Begin Tile parameter	ID (1)	X'8C'	
	LENGTH (1)	X'00'	
Tile Position parameter	ID (1)	X'B5'	
	LENGTH (1)	X'08'	
	XOFFSET (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile origin
	YOFFSET (4)	X'00000000' – X'7FFFFFFF'	Vertical tile origin
Tile Size parameter	ID (1)	X'B6'	
	LENGTH (1)	X'08' – X'09'	
	THSIZE (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile size
	TVSIZE (4)	X'00000000' – X'7FFFFFFF'	Vertical tile size
	RELRES (1)	X'01'	Relative resolution
Tile parameters			

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'03' – X'03'	
	COMPRID (1)	X'01', X'03', X'08', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read (see General note) **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) **X'82'** G4 MMR—Modified Modified Read (see General note)
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR (1)	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01'	1 bit/IDE
IDE Structure parameter	ID (1)	X'9B'	
	LENGTH (1)	X'06' – X'08'	
	FLAGS (1)		
	ASFLAG	B'0'	Additive
	GRAYCODE	B'0'	No gray coding
	RESERVED	B'000000'	Should be zero
	FORMAT (1)	X'02', X'12'	**X'02'** YCrCb **X'12'** YCbCr
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'01'	1 bit/IDE
	SIZE2 (1)	X'00'	0 bit/IDE
	SIZE3 (1)	X'00'	0 bit/IDE
image data	ID (2)	X'FE92'	
	LENGTH (2)	X'0001' – X'FFFF'	
	DATA	Any	IDEs (see Note on the tile-final parameters)
End Tile	ID (1)	X'8D'	

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
	LENGTH (1)	X'00'	
Note on the tile-final parameters: With IDESZ=1 and LUTID=0, IDE value 0 represents an *insignificant* image point, and 1 represents a *significant* image point. The interpretation of this value is determined by the Set Bilevel Image Color parameter or, lacking that, the device default.			
Final parameters in Function Set 42:			
End Image Content	ID (1)	X'93'	
	LENGTH (1)	X'00'	
End Segment	ID (1)	X'71'	
	LENGTH (1)	X'00'	
General note: ABIC, InfoPrint MMR—Modified Modified Read and G4 MMR—Modified Modified READ are applicable only to images whose IDE size is 1 bit per band, otherwise exception condition EC-9611 is raised.			

IOCA Function Set 42 (IOCA FS42)

Function Set 42 is a subset of Function Set 45. It describes tiled images with one bit per spot. Images can be either bilevel (color space YCbCr or YCrCb, IDESZ=1) or color (color space CMYK, IDESZ=4). This function set is carried by the MO:DCA-P controlling environment. The permissible parameter groupings in FS42 are defined as follows:

Table 10. Function Set 42 structure

	X'70'	Begin Segment parameter		
	X'91'	Begin Image Content parameter		
	X'FEBB'	Tile TOC parameter		
[	X'95'	Image Encoding parameter		]
[	X'96'	IDE Size parameter		]
[	X'98'	Band Image parameter		]
[	X'9B'	IDE Structure parameter		]
[		Tiles	(S)	]
	X'93'	End Image Content parameter		
	X'71'	End Segment parameter		

Table 11. Tile structure

	X'8C'	Begin Tile parameter		
	X'B5'	Tile Position parameter		
	X'B6'	Tile Size parameter		
[	X'95'	Image Encoding parameter		]
[	X'96'	IDE Size parameter		]
[	X'98'	Band Image parameter		]
[	X'9B'	IDE Structure parameter		]
[	X'B7'	Tile Set Color parameter		]
[		Image Data or Band Image Data	(S)	]
	X'8D'	End Tile		

Notes:

1. Note that the parameters in Table 10 and Table 11 must come in the specified order. Even though the general IOCA architecture allows different ordering for some of the parameters, the FS42 specification is more restrictive. If the parameters are given in a different order, an out-of-sequence exception is raised.
2. In the context of FS42, Image Size parameter, Image Subsampling parameter, External Algorithm parameter, and Image LUT ID parameter cause EC-0001 exception (Invalid parameter) to occur. If the first parameter after the Begin Image Content parameter is not the Tile TOC parameter, the image is not a tiled image and any of the tile-specific parameters (Tile TOC parameter, Begin Tile parameter, etc.) cause EC-0001 to occur.
3. If the IDE Size is not set to 1 bit or the color space is not YCbCr or YCrCb for a tile, and the Tile Set Color parameter is specified, exception EC-B711 occurs.
4. If the Solid Fill Rectangle compression algorithm is specified for a tile and Image Data or Band Image Data is encountered, exception EC-0001 occurs.
5. Image Encoding parameter, IDE Size parameter, Band Image parameter, and IDE Structure parameter are shown as optional and can possibly be specified in two places. The specification within a tile takes precedence over a specification outside of the tile.
6. If IDE Size parameter is not present, the default IDE size is one bit per pel (bilevel image).

7. If the Image Encoding parameter is not present, the default compression algorithm is X'03' (No Compression). The recording algorithm defaults to X'01' (RIDIC); and the bit and byte orders default to zero.
8. If a tile contains the IDE Structure parameter specifying CMYK color space, then IDE Size parameter, Band Image parameter, and Band Image Data must also be present.
9. If the IDE Structure parameter specifying CMYK color space is given outside of the tiles, then IDE Size parameter and Band Image parameter must be given either outside of the tiles or within every tile that does not contain another IDE Structure parameter specifying that the tile is bilevel.
10. CMYK tiles must carry the image data in Band Image Data. Bilevel tiles must carry the data in Image Data, unless the Solid Fill Rectangle compression algorithm is specified.
11. If a tile has the Solid Fill Rectangle specified as the compression algorithm, the tile is painted using the color specified in the Tile Set Color parameter for that tile. If the Tile Set Color parameter has not been specified, color given using the Set Bilevel Image Color field in the Image Data Descriptor is used. If the Set Bilevel Image Color field is missing, the device default is used.

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Initial parameters in Function Set 42:			
Begin Segment	ID (1)	X'70'	
	LENGTH (1)	X'00'	
Begin Image Content	ID (1)	X'91'	
	LENGTH (1)	X'01'	
	OBJTYPE (1)	X'FF'	IOCA
Tile TOC parameter	ID (2)	X'FEBB'	
	LENGTH (2)		
	Reserved (2)	X'0000'	Reserved; should be set to zero
	Either zero repeating groups, or one per tile in the following format:		
	XOFFSET (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile origin
	YOFFSET (4)	X'00000000' – X'7FFFFFFF'	Vertical tile origin
	THSIZE (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile size
	TVSIZE (4)	X'00000000' – X'7FFFFFFF'	Vertical tile size
	RELRES (1)	X'01'	Relative resolution
	COMPR (1)		Compression algorithm
	DATAPOS (8)		File offset to the beginning of the tile

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'03'	
	COMPRID (1)	X'01', X'03', X'08', X'20', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read (see General note) **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) (see General note) **X'20'** Solid Fill Rectangle **X'82'** G4 MMR—Modified Modified Read (see General note)
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01', X'04'	**X'01'** 1 bit/IDE **X'04'** 4 bits/IDE
Initial parameters in a tile:			
Begin Tile parameter	ID (1)	X'8C'	
	LENGTH (1)	X'00'	
Tile Position parameter	ID (1)	X'B5'	
	LENGTH (1)	X'08'	
	XOFFSET (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile origin
	YOFFSET (4)	X'00000000' – X'7FFFFFFF'	Vertical tile origin

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Tile Size parameter	ID (1)	X'B6'	
	LENGTH (1)	X'08' – X'09'	
	THSIZE (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile size
	TVSIZE (4)	X'00000000' – X'7FFFFFFF'	Vertical tile size
	RELRES (1)	X'01'	Relative resolution
Tile parameters used when IDESZ=1:			
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'03' – X'03'	
	COMPRID (1)	X'01', X'03', X'08', X'20', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read (see General note) **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) **X'20'** Solid Fill Rectangle **X'82'** G4 MMR—Modified Modified Read (see General note)
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR (1)	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01'	1 bit/IDE
Band Image parameter	ID (1)	X'98'	
	LENGTH (1)	X'02'	
	BCOUNT (1)	X'01'	One band
	BITCNT (1)	X'01'	1 bit/IDE

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
IDE Structure parameter	ID (1)	X'9B'	
	LENGTH (1)	X'06' – X'08'	
	FLAGS (1)		
	ASFLAG	B'0'	Additive
	GRAYCODE	B'0'	No gray coding
	RESERVED	B'000000'	Should be zero
	FORMAT (1)	X'02', X'12'	**X'02'** YCrCb **X'12'** YCbCr
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'01'	1 bit/IDE
	SIZE2 (1)	X'00'	0 bit/IDE
	SIZE3 (1)	X'00'	0 bit/IDE
Tile Set Color	ID (1)	X'B7'	
	LENGTH (1)	X'10'	
	CSPACE (1)	X'04', X'08'	**X'04'** CMYK **X'08'** CIELab
	RESERVED (3)	X'000000'	Should be zero
	SIZE1–SIZE3 (1)	X'08'	Bits/IDE for components 1-3
	SIZE4 (1)	X'00', X'08'	Bits/IDE for components 4
	CVAL1–CVAL4 (2)	X'0000' – X'00FF'	Color values
Tile parameters used when IDESZ=4:			

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'03'	
	COMPRID (1)	X'01', X'03', X'08', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read (see General note) **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) **X'82'** G4 MMR—Modified Modified Read (see General note)
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR (1)	X'00', X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'04'	4 bit/IDE
Band Image parameter	ID (1)	X'98'	
	LENGTH (1)	X'05'	
	BCOUNT (1)	X'04'	Four Bands: CMYK
	BITCNT (1)	X'01'	1 bit/IDE for C band
	BITCNT (1)	X'01'	1 bit/IDE for M band
	BITCNT (1)	X'01'	1 bit/IDE for Y band
	BITCNT (1)	X'01'	1 bit/IDE for K band

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
IDE Structure parameter	ID (1)	X'01'	
	LENGTH (1)	X'01'	
	FLAGS (1)		
	ASFLAG	B'0'	Additive
	GRAYCODE	B'0'	No gray coding
	RESERVED	B'000000'	Should be zero
	FORMAT (1)	X'04'	CMYK
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'01'	1 bit/IDE (C component)
	SIZE2 (1)	X'01'	1 bit/IDE (M component)
	SIZE3 (1)	X'01'	1 bit/IDE (Y component)
	SIZE4 (1)	X'01'	1 bit/IDE (K component)
Final parameters in a tile:			
image data	ID (2)	X'FE92'	
	LENGTH (2)	X'0001' – X'FFFF'	
	DATA	Any	IDEs (see Note on the tile-final parameters)
or			
band image data (BCOUNT=4 only)	**Four bands, in order by BANDNUM, in the following format:**		
	ID (2)	X'FE9C'	
	LENGTH (2)	X'0004' – X'FFFF'	
	BANDNUM (1)	X'01' – X'04'	**X'01'** Band contains the C component of the IDEs **X'02'** Band contains the M component of the IDEs **X'03'** Band contains the Y component of the IDEs **X'04'** Band contains the K component of the IDEs
	RESERVED (2)	X'0000'	Should be zero
	DATA	Any	
End Tile	ID (1)	X'8D'	
	LENGTH (1)	X'00'	

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Note on the tile-final parameters: With IDESZ=1 and LUTID=0, IDE value 0 represents an *insignificant* image point, and 1 represents a *significant* image point. The interpretation of this value is determined by the Tile Set Color parameter, if one has been specified. Otherwise, the color specified via Set Bilevel Image Color parameter applies or, lacking that, the device default.			
Final parameters in Function Set 42:			
End Image Content	ID (1)	X'93'	
	LENGTH (1)	X'00'	
End Segment	ID (1)	X'71'	
	LENGTH (1)	X'00'	
General note: ABIC, InfoPrint MMR—Modified Modified Read, G4 MMR—Modified Modified READ, and Solid Fill Rectangle are applicable only to images whose IDE size is 1 bit per band, otherwise exception condition EC-9611 is raised.			

IOCA Function Set 45 (IOCA FS45)

Function Set 45 is a superset of Function Set 42. It describes bilevel or color tiled images. This function set is carried by the MO:DCA-P controlling environment. The permissible parameter groupings in FS45 are now defined as follows:

Table 12. Function Set 45 structure

	X'70'	Begin Segment parameter		
		Image Content	(S)	
	X'71'	End Segment parameter		

Table 13. Image Content structure

	X'91'	Begin Image Content parameter		
	X'FEBB'	Tile TOC parameter		
[	X'95'	Image Encoding parameter		]
[	X'96'	IDE Size parameter		]
[	X'98'	Band Image parameter		]
[	X'9B'	IDE Structure parameter		]
[		Data and Referencing Tiles	(S)	]
	X'93'	End Image Content parameter		

Table 14. Data Tile structure

	X'8C'	Begin Tile parameter		
	X'B5'	Tile Position parameter		
	X'B6'	Tile Size parameter		
[	X'95'	Image Encoding parameter		]
[	X'96'	IDE Size parameter		]
[	X'98'	Band Image parameter		]
[	X'9B'	IDE Structure parameter		]
[	X'9F'	External Algorithm Specification parameter (ignored)		]
[	X'B7'	Tile Set Color parameter		]
[		Transparency Mask		]
[		Image Data or Band Image Data	(S)	]
	X'8D'	End Tile		

Table 15. Referencing Tile structure

	X'8C'	Begin Tile parameter		
	X'B5'	Tile Position parameter		
	X'B6'	Include Tile parameter		
[		Transparency Mask		]
	X'8D'	End Tile		

Table 16. IOCA Tile Resource structure

	X'8C'	Begin Tile parameter		
	X'B5'	Tile Position parameter		
	X'B6'	Tile Size parameter		
[	X'95'	Image Encoding parameter		]
[	X'96'	IDE Size parameter		]
[	X'98'	Band Image parameter		]
[	X'9B'	IDE Structure parameter		]
[	X'9F'	External Algorithm Specification parameter (ignored)		]
[	X'B7'	Tile Set Color parameter		]
[		Transparency Mask		]
[		Image Data or Band Image Data	(S)	]

Table 16. IOCA Tile Resource structure (continued)

	X'8D'	End Tile	

Table 17. Transparency Mask structure

	X'8E'	Begin Transparency Mask parameter	
	X'94'	Image Size parameter	
[	X'95'	Image Encoding parameter	]
	X'FE92'	Image Data	
	X'8F'	End Transparency Mask parameter	

Notes:

1. Note that the parameters in Table 12 on page 118, Table 13 on page 118, Table 14 on page 118, Table 15 on page 118, Table 16 on page 118, and Table 17 must come in the specified order. Even though the general IOCA architecture allows different ordering for some of the parameters, the FS45 specification is more restrictive. If the parameters are given in a different order, an out-of-sequence exception is raised.
2. Image Encoding parameter, IDE Size parameter, Band Image parameter, and IDE Structure parameter are shown as optional and can possibly be specified in two places. Note that tile data may require that some of these parameters be specified.
3. If IDE Size parameter is not present neither in the tile nor in the image content, default IDE size is one bit per pel (bilevel image).
4. If the Image Encoding parameter is not present, the default compression algorithm is X'03' (No Compression); the recording algorithm defaults to X'01' (RIDIC); and the bit and byte orders default to zero.
5. If a tile contains the IDE Structure parameter specifying CMYK color space, then IDE Size parameter, Band Image parameter, and Band Image Data must also be present.
6. If the IDE Structure parameter specifying CMYK color space is given outside of the tiles, then IDE Size parameter and Band Image parameter must be given either outside of the tiles or within every tile that does not contain another IDE Structure parameter specifying a different color space.
7. Receivers implementing FS45 must support at least 128 image contents in a single image segment. Otherwise, if a receiver encounters too many image contents to process, it should act as if it encountered too many image objects on a page.
8. Resource tiles included via Include Tile parameter must not contain Include Tile parameter or the exception EC-B811 exists.

The self-defining fields and parameter values that are acceptable in Function Set 45 are shown in the following table.

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Initial parameters in Function Set 45:			
Begin Segment	ID (1)	X'70'	
	LENGTH (1)	X'00'	
Begin Image Content	ID (1)	X'91'	
	LENGTH (1)	X'01'	
	OBJTYPE (1)	X'FF'	IOCA

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Tile TOC parameter	ID (2)	X'FEBB'	
	LENGTH (2)		
	Reserved (2)	X'0000'	Reserved; should be set to zero
	Either zero repeating groups, or one per tile in the following format:		
	XOFFSET (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile origin
	YOFFSET (4)	X'00000000' – X'7FFFFFFF'	Vertical tile origin
	THSIZE (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile size
	TVSIZE (4)	X'00000000' – X'7FFFFFFF'	Vertical tile size
	RELRES (1)	X'01' – X'02'	Relative resolution (see Note 1)
	COMPR (1)		Compression algorithm
	DATAPOS (8)		File offset to the beginning of the tile
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'02'	
	COMPRID (1)	X'01', X'03', X'08', X'0D', X'20', X'82' – X'83'	**X'01'** InfoPrint MMR—Modified Modified Read (see Note 2) **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) (see Note 2) **X'0D'** TIFF LZW **X'20'** Solid Fill Rectangle **X'82'** G4 MMR—Modified Modified Read (see Note 2) **X'83'** JPEG algorithms (see Note 3)
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR (1)	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01', X'04', X'20'	**X'01'** 1 bit/IDE **X'04'** 4 bits/IDE **X'20'** 32 bits/IDE
Notes on the initial parameters: 1. In the Tile TOC parameter, the relative resolution (RELRES) of 2 is supported only for JPEG-compressed data. Using RELRES of 1 for JPEG-compressed data and RELRES of 2 for non-JPEG data results in exception EC-B610 being raised. Note that this restriction on the relative resolution holds only for this function set, not for the IOCA architecture in general. 2. ABIC (Bilevel Q-Coder), InfoPrint MMR—Modified Modified Read, G4 MMR—Modified Modified READ and Solid Fill Rectangle are applicable only to images whose IDE size is 1 bit/band, otherwise exception condition EC-9611 is raised. 3. The JPEG algorithms are applicable only to images whose IDE size is 32 bits/IDE; otherwise exception condition EC-9611 is raised.			
Initial parameters in a data tile:			
Begin Tile parameter	ID (1)	X'8C'	
	LENGTH (1)	X'00'	
Tile Position parameter	ID (1)	X'B5'	
	LENGTH (1)	X'08'	
	XOFFSET (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile origin
	YOFFSET (4)	X'00000000' – X'7FFFFFFF'	Vertical tile origin
Tile Size parameter	ID (1)	X'B6'	
	LENGTH (1)	X'08' – X'09'	
	THSIZE (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile size
	TVSIZE (4)	X'00000000' – X'7FFFFFFF'	Vertical tile size
	RELRES (1)	X'01' – X'02'	Relative resolution (see Note on the data-tile initial parameters)
Note on the data-tile initial parameters: In the Tile Size parameter, the relative resolution (RELRES) of 2 is supported only for JPEG-compressed data. Using RELRES of 1 for JPEG-compressed data and RELRES of 2 for non-JPEG data results in exception EC-B610 being raised. Note that this restriction on the relative resolution holds only for this function set, not for the IOCA architecture in general.			
Initial parameters in a referencing tile:			
Begin Tile parameter	ID (1)	X'8C'	
	LENGTH (1)	X'00'	

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Tile Position parameter	ID (1)	X'B5'	
	LENGTH (1)	X'08'	
	XOFFSET (4)	X'00000000' – X'7FFFFFFF'	Horizontal tile origin
	YOFFSET (4)	X'00000000' – X'7FFFFFFF'	Vertical tile origin
Include Tile parameter	ID (2)	X'FEB8'	
	LENGTH (1)	X'08'	
	TIRID (4)	X'00000000' – X'FFFFFFFF'	Resource Tile local identifier
Parameters used when IDESZ=1:			
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'03'	
	COMPRID (1)	X'01', X'03', X'08', X'20', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) **X'20'** Solid Fill Rectangle **X'82'** G4 MMR—Modified Modified Read
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR (1)	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'01'	1 bit/IDE
Band Image parameter	ID (1)	X'98'	
	LENGTH (1)	X'02'	
	BCOUNT (1)	X'01'	One band
	BITCNT (1)	X'01'	1bit/IDE

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
IDE Structure parameter	ID (1)	X'9B'	
	LENGTH (1)	X'06' – X'08'	
	FLAGS (1)		
	ASFLAG	B'0'	Additive
	GRAYCODE	B'0'	No gray coding
	RESERVED	B'00000'	Should be zero
	FORMAT (1)	X'02', X'12'	**X'02'** YCrCb **X'12'** YCbCr
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'01'	1 bit/IDE
	SIZE2 (1)	X'00'	0 bit/IDE
	SIZE3 (1)	X'00'	0 bit/IDE
Tile Set Color	ID (1)	X'B7'	
	LENGTH (1)	X'1'	
	CSPACE (1)	X'04', X'08'	**X'04'** CMYK **X'08'** CIELab
	RESERVED (3)	X'000000'	Should be zero
	SIZE1–SIZE3 (1)	X'08'	Bits/IDE for components 1-3
	SIZE4 (1)	X'00', X'08'	Bits/IDE for components 4
	CVAL1–CVAL4 (2)	X'0000' – X'00FF'	Color values
Parameters used when IDESZ=4:			
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'03'	
	COMPRID (1)	X'01', X'03', X'08', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) **X'82'** G4 MMR—Modified Modified Read
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR (1)	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'04'	4 bit/IDE
Band Image parameter	ID (1)	X'98'	
	LENGTH (1)	X'05'	
	BCOUNT (1)	X'04'	Four Bands: CMYK
	BITCNT (1)	X'01'	1 bit/IDE for C band
	BITCNT (1)	X'01'	1 bit/IDE for M band
	BITCNT (1)	X'01'	1 bit/IDE for Y band
	BITCNT (1)	X'01'	1 bit/IDE for K band
IDE Structure parameter	ID (1)	X'01'	
	LENGTH (1)	X'01' – X'01'	
	FLAGS (1)		
	ASFLAG	B'0'	Additive
	GRAYCODE	B'0'	No gray coding
	RESERVED	B'000000'	Should be zero
	FORMAT (1)	X'04'	CMYK
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'01'	1 bit/IDE (C component)
	SIZE2 (1)	X'01'	1 bit/IDE (M component)
	SIZE3 (1)	X'01'	1 bit/IDE (Y component)
	SIZE4 (1)	X'01'	1 bit/IDE (K component)
Parameters used when IDESZ=32:			
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'03'	
	COMPRID (1)	X'03', X'0D', X'83'	**X'03'** No Compression **X'0D'** TIFF LZW **X'83'** JPEG algorithms
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR (1)	X'00' – X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left
IDE Size parameter	ID (1)	X'96'	
	LENGTH (1)	X'01'	
	IDESZ (1)	X'20'	32 bit/IDE

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Band Image parameter	ID (1)	X'98'	
	LENGTH (1)	X'05'	
	BCOUNT (1)	X'04'	4 bands: CMYK
	BITCNT (1)	X'08'	8 bit/IDE for C band
	BITCNT (1)	X'08'	8 bit/IDE for M band
	BITCNT (1)	X'08'	8 bit/IDE for Y band
	BITCNT (1)	X'08'	8 bit/IDE for K band
IDE Structure parameter	ID (1)	X'9B'	
	LENGTH (1)	X'09'	
	FLAGS (1)		
	SFLAG	B'0'	Additive
	GRAYCODE	B'0'	No gray coding
	RESERVED	B'000000'	Should be zero
	FORMAT (1)	X'04'	CMYK
	RESERVED (3)	X'000000'	Should be zero
	SIZE1 (1)	X'08'	8 bits/IDE (C component)
	SIZE2 (1)	X'08'	8 bits/IDE (M component)
	SIZE3 (1)	X'08'	8 bits/IDE (Y component)
	SIZE4 (1)	X'08'	8 bits/IDE (K component)
Final parameters in a tile:			
Begin Transparency Mask	ID (1)	X'8E'	
	LENGTH (1)	X'00'	
Image Size parameter	ID (1)	X'94'	**X'00'** 10 inches **X'01'** 10 centimeters
	LENGTH (1)	X'09'	
	UNITBASE (1)	X'00' – X'01'	
	HRESOL (2)	X'0001' – X'7FFF'	
	VRESOL (2)	X'0001' – X'7FFF'	
	HSIZE (2)	X'0001' – X'7FFF'	
	VSIZE (2)	X'0001' – X'7FFF'	

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Image Encoding parameter	ID (1)	X'95'	
	LENGTH (1)	X'02' – X'03'	
	COMPRID (1)	X'01', X'03', X'08', X'82'	**X'01'** InfoPrint MMR—Modified Modified Read **X'03'** No Compression **X'08'** ABIC (Bilevel Q-Coder) **X'82'** G4 MMR—Modified Modified Read
	RECID (1)	X'01', X'04'	**X'01'** RIDIC **X'04'** Unpadded RIDIC
	BITORDR (1)	X'00', X'01'	**X'00'** Bit order within each image data byte is from left to right **X'01'** Bit order within each image data byte is from right to left
image data	ID (2)	X'FE92'	
	LENGTH (2)	X'0001' – X'FFFF'	
	DATA	Any	IDEs (bilevel only)
End Transparency Mask	ID (1)	X'8F'	
image data	LENGTH (1)	X'8F'	
	ID (2)	X'00'	
	LENGTH (2)	X'0001' – X'FFFF'	
	DATA	Any	IDEs (see Note 1)
or:			

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
band image data (BCOUNT=4 only)	**Four bands, in order by BANDNUM, in the following format:**		
	ID (2)	X'FE9C'	
	LENGTH (2)	X'0003' – X'FFFF'	(see Note 2)
	BANDNUM (1)	X'01' – X'04'	**X'01'** Band contains the C component of the IDEs **X'02'** Band contains the M component of the IDEs **X'03'** Band contains the Y component of the IDEs **X'04'** Band contains the K component of the IDEs
	RESERVED (2)	X'0000'	Should be zero
	DATA	Any	
End Tile	ID (1)	X'8D'	
	LENGTH (1)	X'00'	

Notes on the tile-final parameters:

1. With IDESZ=1 and LUTID=0, IDE value 0 represents an *insignificant* image point, and 1 represents a *significant* image point. The interpretation of this value is determined by the Tile Set Color parameter, if one has been specified. Otherwise, the color specified via Set Bilevel Image Color parameter applies or, lacking that, the device default.
2. Band Image Data parameters with length of X'0003' do not have a data field. The receiver generates zeroes for the corresponding band.

IOCA self-defining field	Parameter (bytes)	Acceptable value	Comments
Final parameters in Function Set 45:			
End Image Content	ID (1)	X'93'	
	LENGTH (1)	X'00'	
End Segment	ID (1)	X'71'	
	LENGTH (1)	X'00'	

Appendix A. Compression and recording algorithms

This appendix describes in more detail the image compression and recording algorithms currently supported by the IOCA Image Encoding parameter.

This chapter consists of the image compression and recording algorithms which are presently defined in "Image Encoding" on page 33. This appendix is not meant to be a complete description or specification of each algorithm, but is meant to be a short and concise outline of the characteristics of each image compression algorithm.

Compression algorithms

The following compression algorithms are described in this document. The number to the left of each algorithm is the value which the compression algorithm represents for the COMPRID parameter of the Image Encoding parameter.

Value	Algorithm
X'01'	InfoPrint MMR—Modified Modified Read
X'03'	No compression
X'06'	RL4 (Run Length 4)
X'08'	ABIC (Bilevel Q-Coder)
X'0A'	Concatenated ABIC
X'0B'	Color compression used by OS/2 Image Support, part number 49F4608
X'0C'	TIFF PackBits
X'0D'	TIFF LZW
X'20'	Solid Fill Rectangle
X'80'	G3 MH—Modified Huffman (ITU–TSS T.4 Group 3 one-dimensional coding standard for facsimile)
X'81'	G3 MR—Modified READ (ITU–TSS T.4 Group 3 two-dimensional coding option for facsimile)
X'82'	G4 MMR—Modified Modified READ (ITU–TSS T.6 Group 4 two-dimensional coding standard for facsimile)
X'83'	JPEG algorithms (see the External Algorithm Specification parameter for detail)
X'84'	JBIG2
X'FE'	User-defined algorithms (see the External Algorithm Specification parameter for details)
Other values	All other values are reserved.

All of these compression algorithms are *lossless* (they result in no loss of data) except for some JPEG algorithms, which are *lossy*.

Modified ITU–TSS Modified READ algorithm (InfoPrint MMR—Modified Modified Read)

This compression algorithm was developed by modifying the ITU–TSS Modified READ (Relative Element Algorithm Designate) algorithm. It allows both one- and two-dimensional correlations among changing color points in image data:

- In one-dimensional (1D) mode, color transitions in the image are coded by a run-length that denotes how long the color continues in the horizontal direction.
- In two-dimensional (2D) mode, the image is coded by how far each IDE is positioned from different color IDEs on the same line or the previous line.

The InfoPrint MMR—Modified Modified Read algorithm differs from the ITU–TSS Modified READ algorithm in the following aspects:
- Infinite K value (only the first scan line is in 1D mode)
- No EOLs, except when switching from 1D to 2D and as part of the EOP
- No time-fill bit

The InfoPrint MMR—Modified Modified Read algorithm also differs from a related algorithm, the ITU–TSS Modified Modified READ algorithm, in that the InfoPrint MMR—Modified Modified Read uses one-dimensional coding for the first image line and two-dimensional coding for the remaining lines, while the ITU–TSS Modified Modified READ algorithm uses two-dimensional coding only.

With the Modified ITU–TSS Modified READ algorithm, only one EOP appears at the end of image content.

Notes:
1. InfoPrint MMR—Modified Modified Read allows the IOCA process model to determine the number of image points in the horizontal and vertical directions. HSIZE and VSIZE can therefore be zero in the Image Size parameter.
2. If the HSIZE or VSIZE parameter of the Image Size parameter is non-zero, it may be less than the actual number of horizontal or vertical image points encoded in the image data due to padding bits or padding scan lines.

For more details about the ITU–TSS Modified READ algorithm, refer to *Standardization of Group 3 Facsimile Apparatus for Document Transmission*, ITU–TSS Recommendation T.4.

For more details about the ITU–TSS Modified Modified READ algorithm, refer to *Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus*, ITU–TSS Recommendation T.6.

For more details about the InfoPrint MMR—Modified Modified Read compression algorithm, refer to “Binary-image-manipulation Algorithms in the Image View Facility” in *IBM Journal of Research and Development*, Volume 31, Number 1 (January 1987).

No Compression

This method sends raw image data, in binary form, without any reduction.

Note: The value No Compression does *not* allow the IOCA process model to determine the number of horizontal image points from the image data. However, VSIZE can be zero in the Image Size parameter.

Run Length 4 (RL4) compression algorithm

The Run Length 4 (RL4) algorithm is a binary, one-dimensional, run-length coding method of compression. It is based on code words using four bits. The code words used are common to both white runs and black runs. Table 18 lists the code words.

Table 18. RL4 code words

Run Length	Code Word	Code Length
0	B'1111 1110'	8 bits
1–8	B'0'*xxx*	4 bits
9–72	B'10'*xx xxxx*	8 bits
73–584	B'110'*x xxxx xxxx*	12 bits
585–4680	B'1110'*xxxx xxxx xxxx*	16 bits
4681–32767	B'1111 0'*xxx xxxx xxxx xxxx*	20 bits
EOL	B'1111 1111 (1111)'	8 or 12 bits

Two EOL (End-Of-Line) codes are provided to make an encoded string of each scan line start at a byte boundary. Either of these codes is used, depending on whether or not the last run-length code of the previous scan line ends at a byte boundary. Each scan line is represented in the following format:

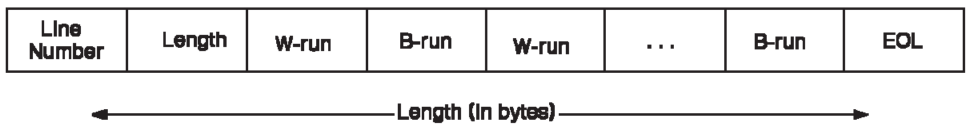


Figure 18. Scan line format

Both line number and length are represented as two-byte integers, making it possible to skip lines efficiently or to access a specific line directly for display and editing purposes. Providing line numbers also allows completely white lines to be skipped when recording the compressed data.

Regarding the run encoding, the first run of each line must be white; if a line begins with a black image data element, a white run of length zero must be put in the encoded string. If a line ends with a sequence of white image data elements (which is often the case), the last white run need not be encoded, because it can be calculated from the horizontal size of the image content and the total length of the preceding runs.

Note: RL4 does *not* allow the IOCA process model to determine the number of horizontal image points from the image data. However, VSIZE can be zero in the Image Size parameter.

ABIC (Bilevel Q-Coder) compression algorithm

This algorithm uses an arithmetic coding technique to produce *lossless data compression*, which is an invertible mapping between any data file and a compact representation of the same information.

Note: ABIC does *not* allow the IOCA process model to determine the number of horizontal or vertical image points from the image data. Hence both HSIZE and VSIZE cannot be zero in the Image Size parameter.

For more details, refer to R. Arps, T. Truong, D. Lu, R. Pasco, and T. Friedman, "A multipurpose VLSI chip for adaptive data compression of bilevel images", in *IBM Journal of Research and Development*, Volume 32, No. 6 (November 1988).

TIFF algorithm 2 compression algorithm

Tag Image File Format (TIFF) data compression scheme 2 is a method of compression that enables image data to be compressed one-dimensionally and is based upon the ITU–TSS T.4 G3 facsimile one-dimensional coding scheme (G3 MH—Modified Huffman).

The TIFF data compression scheme 2 differs from the ITU–TSS T.4 G3 facsimile one-dimensional coding scheme (G3 MH—Modified Huffman) in the following respects:

- New rows always begin on the next available byte boundary.
- No End-of-line (EOL) code words are used.
- No fill bits are used, except for the ignored bits at the end of the last byte of a row.
- No Return to control (RTC) is used.

Note: TIFF 2 does *not* allow the IOCA process model to determine the number of horizontal or vertical image points from the image data. Hence both HSIZE and VSIZE cannot be zero in the Image Size parameter.

For more details about the ITU–TSS Group 3 algorithms, refer to *Standardization of Group 3 Facsimile Apparatus for Document Transmission*, ITU–TSS Recommendation T.4.

Concatenated ABIC compression algorithm

This algorithm is a form of compression that utilizes the ABIC compression algorithm.

For image data with an IDE size of *n* bits, a processor begins the compression process by retrieving the first bit of the first IDE, and continuing until the first bit of every IDE has been retrieved, in the order in which the IDEs were recorded. The processor then retrieves the second bit of the first IDE, and so on until all the second bits have been retrieved. This sequential process is continued until the *n*th bit of every IDE has been retrieved.

The raster data obtained from this process is compressed using the ABIC algorithm to form a single string of ABIC compressed image data. This compression may occur during the retrieval process, or after all the raster data has been retrieved. No break in the code indicating an End-of-Line, End-of-Page, or a flag may appear in the compressed data. Thus, the length of each line, the size of the image, and the number of bits per IDE cannot be determined from the concatenated ABIC compressed image data.

Note: Concatenated ABIC does *not* allow the IOCA process model to determine the number of horizontal or vertical image points from the image data. Hence both HSIZE and VSIZE cannot be zero in the Image Size parameter.

For more details about the concatenated ABIC algorithm, refer to Arps et al., "A multipurpose VLSI chip for adaptive data compression of bilevel images".

OS/2 Image Support compression algorithm

The color compression algorithm supported by the OS/2 Image Support program, part number 49F4608, is based on an earlier revision (R5.0) of the JPEG draft specification. It is similar to the JPEG baseline system described in "JPEG compression algorithms" on page 135.

The OS/2 Image Support program supports data streams in RGB pixel interleaf format only: that is, the color pixels input to the encoder and the decoder output must be of the form *RGBRGB*...*RGB*.

Note: The OS/2 Image Support implementation of the JPEG compression algorithm does *not* allow the IOCA process model to determine the number of horizontal or vertical image points from the image data. Hence both HSIZE and VSIZE cannot be zero in the Image Size parameter.

For more details, refer to William B. Pennebaker and Joan L. Mitchell, "Standardization of Color Image Data Compression", Part I. "Sequential Coding", in *Proceedings Electronic Imaging '89 East* (October 2–5, 1989): 109–112.

TIFF PackBits compression algorithm

The TIFF PackBits algorithm came from the Apple Macintosh system and is applicable to bilevel images only. Each line is packed independently of any other lines.

Note: TIFF PackBits does *not* allow the IOCA process model to determine the number of horizontal or vertical image points from the image data. Hence both HSIZE and VSIZE cannot be zero in the Image Size parameter.

For more details about the algorithm, refer to *TIFF*™, Revision 6.0, Final (Aldus Corp.: June 3, 1992).

TIFF LZW compression algorithm

The LZW (Lempel-Ziv and Welch) algorithm is applicable to bilevel, and continuous tone or palette grayscale and color images. The algorithm works best if the input uncompressed data is organized into strips of about 8K bytes, with each strip being compressed independently.

The algorithm is based on a translation table, or string table, that maps strings of input characters into codes. The TIFF implementation uses variable-length codes, with a maximum code length of twelve bits. This string table is different for every strip.

Notes:

1. TIFF LZW does *not* allow the IOCA process model to determine the number of horizontal or vertical image points from the image data. Hence both HSIZE and VSIZE cannot be zero in the Image Size parameter.
2. LZW encoders sometimes terminate the data early. If the LZW decoder does not produce the expected number of bytes, no exception should be raised and the receiver should fill the remaining data with binary zeroes.

For more details about the algorithm, refer to:

- *TIFF*™, Revision 6.0, Final (Aldus Corp.: June 3, 1992).
- Terry A. Welch, "A Technique for High Performance Data Compression", in *IEEE Computer*, vol. 17, no. 6 (June 1984).

Solid Fill Rectangle compression algorithm

The Solid Fill Rectangle compression algorithm is applicable to tiled images only. It indicates that the tile contains no image data (Image Data or Band Image Data). Instead, the tile may contain Tile Set Color parameter and all the image points contained within the tile are painted by the color specified in the Tile Set Color parameter. If the Tile Set Color parameter is missing, the color is specified via the Set Bilevel Image Color parameter. If Set Bilevel Image Color is missing, the device default color is used. The Solid Fill Rectangle compression algorithm is applicable only in bilevel color spaces (IDESZ=1), since Tile Set Color specifies the color space internally and requires that the tile color space specified via the optional IDE Structure and IDE Size parameters be bilevel (that is, as either YCbCr or YCrCb and with the IDE Size as 1).

Since the tile compressed using the Solid Fill Rectangle algorithm is generated by the receiver based on the tile dimensions, the THSIZ and TVSIZE fields in the Tile Size Parameters must both be nonzero.

ITU–TSS T.4 Group 3 Coding Standard (G3 MH—Modified Huffman) for Facsimile

The ITU–TSS T.4 Group 3 Coding Standard (G3 MH—Modified Huffman) is a compression method for facsimile, standardized by the ITU–TSS (formerly CCITT). It enables one-dimensional compression.

Note: G3 MH—Modified Huffman does *not* allow the IOCA process model to determine the number of image points in the horizontal direction. However, VSIZE can be zero in the Image Size parameter.

For more details, refer to *Standardization of Group 3 Facsimile Apparatus for Document Transmission*, ITU–TSS Recommendation T.4.

ITU–TSS T.4 Group 3 Coding Option (G3 MR—Modified READ) for Facsimile

The ITU–TSS T.4 Group 3 Coding Option (G3 MR—Modified READ) is a compression method for facsimile, standardized by the ITU–TSS. It enables two-dimensional compression.

Note: G3 MR—Modified READ does *not* allow the IOCA process model to determine the number of image points in the horizontal direction. However, VSIZE can be zero in the Image Size parameter.

For more details, refer to *Standardization of Group 3 Facsimile Apparatus for Document Transmission*, ITU–TSS Recommendation T.4.

ITU–TSS T.6 Group 4 Coding Standard (G4 MMR—Modified Modified READ) for Facsimile

The ITU–TSS T.6 Group 4 Coding Standard (G4 MMR—Modified Modified READ) is a compression method for facsimile, standardized by the ITU–TSS. It enables two-dimensional compression.

Note: G4 MMR—Modified Modified READ does *not* allow the IOCA process model to determine the number of image points in the horizontal direction. However, VSIZE can be zero in the Image Size parameter.

For more details, refer to *Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus*, ITU–TSS Recommendation T.6.

JPEG compression algorithms

The JPEG (Joint Photographic Experts Group) technical specification details a series of algorithms that can be applied to arbitrary source image resolutions, many color models, multiple image components, various sampling formats, and continuous-tone renditions of text. The algorithms are not applicable to bilevel images.

Some of these algorithms are lossy.

Note: JPEG stores the actual image size in its header, thus allowing the IOCA process model to determine the number of horizontal and vertical image points from the image data. HSIZE and VSIZE can therefore be zero in the Image Size parameter.

For more details, refer to the following publications:
- ISO/IEC International Standard 10918-1
- ITU–TSS Recommendation T.81

JBIG2 (Joint Bi-level Image Experts Group) compression algorithm

JBIG2 embodies a set of techniques for compressing bilevel images. It is generally an asymmetric algorithm in the sense that the compression is more computationally demanding than decompression. The data can be encoded either losslessly, so that the decompressed output is an exact copy of the original, or via lossy algorithms, where the decompressed image is "close" to the original.

JBIG2 is organized as a toolkit of many different algorithms. Different subset of the standard are used for different images and applications. The standard codifies these subsets as “profiles”, much in the same way IOCA uses function sets. JBIG2 receivers commonly implement one or more profiles, and not the whole standard.

JBIG2 compression is defined by the ITU–T Recommendation T.88, ISO/EC 14492:2000 and ITU–T Recommendation T.89 Amendment 1.

Note: JBIG2 stores the actual image size in the compressed datastream, thus allowing the IOCA process model to determine the number of horizontal and vertical image points from the image data. HSIZE and VSIZE can therefore be zero in the Image Size parameter.

For more details, refer to the following publications:
- International Telecommunication Union, Recommendation T.88, “Information Technology—Coded representation of picture and audio information—Lossy/ lossless coding of bi-level images”
- International Telecommunication Union, Recommendation T.89 Amendment 1, “Application Profiles for Recommendation T.88—Lossy/lossless coding of bi-level images (JBIG2) for facsimile”

User-defined compression algorithm

Code point X'FE' denotes that the compression algorithm is supplied by the user, and that the algorithmic description can be found in the External Algorithm

Specification parameter. Users should contact the IOCA architecture group to obtain a unique identifier for their exclusive use.

Compression algorithms and explicit image dimensions

IOCA generators should set HSIZE and VSIZE to the actual width and height of the image, regardless of the compression algorithm used. Leaving either HSIZE or VSIZE as zero may cause some IOCA receivers to abort prematurely. Table 19 lists all the image compression methods recognized by IOCA. The **HSIZE** and **VSIZE** columns are intended for the use of IOCA receivers. Some compression algorithms, such as the InfoPrint MMR—Modified Modified Read, are able to determine the uncompressed image horizontal/vertical size from the input compressed image data, without referring to the HSIZE/VSIZE of the Image Size parameter, that is, HSIZE/VSIZE can be zero in the Image Size parameter.

Table 19. Image compression algorithms supported by IOCA

Compression	COMPRID	HSIZE	VSIZE
InfoPrint MMR—Modified Modified Read	X'01'	Can be zero in the Image Size parameter	Can be zero in the Image Size parameter
No compression	X'03'	Must be non-zero in the Image Size parameter	Can be zero in the Image Size parameter
Run-Length 4	X'06'	Must be non-zero in the Image Size parameter	Can be zero in the Image Size parameter
ABIC (Bilevel Q-Coder)	X'08'	Must be non-zero in the Image Size parameter	Must be non-zero in the Image Size parameter
Concatenated ABIC	X'0A'	Must be non-zero in the Image Size parameter	Must be non-zero in the Image Size parameter
OS/2 Image Support	X'0B'	Must be non-zero in the Image Size parameter	Must be non-zero in the Image Size parameter
TIFF PackBits	X'0C'	Must be non-zero in the Image Size parameter	Must be non-zero in the Image Size parameter
TIFF LZW	X'0D'	Must be non-zero in the Image Size parameter	Must be non-zero in the Image Size parameter
Solid Fill Rectangle	X'20'	Must be non-zero in the Image Size parameter	Must be non-zero in the Image Size parameter
G3 MH—Modified Huffman	X'80'	Must be non-zero in the Image Size parameter	Can be zero in the Image Size parameter
G3 MR—Modified READ	X'81'	Must be non-zero in the Image Size parameter	Can be zero in the Image Size parameter
G4 MMR—Modified Modified READ	X'82'	Must be non-zero in the Image Size parameter	Can be zero in the Image Size parameter
JPEG algorithms	X'83'	Can be zero in the Image Size parameter	Can be zero in the Image Size parameter
JBIG2	X'84'	Can be zero in the Image Size parameter	Can be zero in the Image Size parameter

2. The OS/2 Image Support compression algorithm is based on an earlier version (V5.0) of the JPEG specification. Although JPEG encodes the actual image width and height in the compressed data header the current OS/2 Image Support implementation of the compression algorithm requires both the HSIZE and VSIZE parameters of the Image Size parameter to contain the actual image size.

Compression algorithms for different image types

Each compression algorithm is generally valid for only some of the possible image data types. In some cases, even though the use of a particular algorithm is valid, the compression performance may be poor. Table 20 defines which compression algorithms can be used for each data type.

Table 20. Valid compression algorithms for each data type

IDE size	Color space	Valid algorithms
1 bit/IDE	YCrCb (X'02') YCbCr (X'12')	InfoPrint® MMR (X'01') ABIC (X'08') G4 MMR (X'81')
4 bit/IDE	CMYK (X'04')	InfoPrint MMR (X'01') ABIC (X'08') G4 MMR (X'81')
32 bit/IDE	CMYK (X'04')	TIFF LZW (X'0D') JPEG (X'83')

Notes:

1. Color space is the FORMAT field in the IDE Structure parameter.
2. The compression algorithm is the COMPRID field in the Image Encoding parameter.
3. "No Compression" (X'03') is valid for all image data types.
4. A mismatch between the image data and compression algorithm causes exception EC-9511 to be raised.

The choice of the compression algorithm can have a major impact on both the printer performance and the print quality. Poor compression ratios can result in large datasets that cannot be downloaded to the printer quickly enough. Time required for decompression generally increases with the size of the compressed image and can also be a problem. The print quality is affected by using a lossy algorithm, such as JPEG, on unsuitable data. For more information on matching the compression algorithm to the type of image data, see Appendix F, "Notes for IOCA generators," on page 171.

Recording algorithms

Recording algorithms describe the format of the image data when it is first created. They describe such characteristics as the direction that the IDEs are recorded, and any boundary or formatting constraint that is placed on the image data. The compression takes place on the recorded image.

The recording algorithms that can be specified by the RECID parameter of the Image Encoding parameter are:

Value	**Algorithm**
X'01'	RIDIC (Recording Image Data Inline Coding)
X'03'	Bottom-to-Top
X'04'	Unpadded RIDIC
X'FE'	See the External Algorithm Specification parameter for details.
Other values	All other values are reserved.

RIDIC recording algorithm

The Recorded Image Data Inline Coding (RIDIC) recording algorithm formats the image data as a sequence of binary elements that are generated by the unidirectional raster scan operation. The scanning is from left to right (X direction) and from top to bottom (Y direction), as shown in Figure 19. There are no interlaced fields between the parallel scan lines.

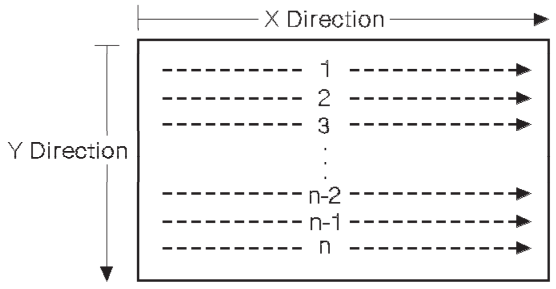


Figure 19. RIDIC recording algorithm

Each raster scan line is in multiples of eight bits. If the width of the image is not a multiple of eight, the scan line must be padded with zeros.

If the ISP specifies a non-multiple of 8 bits, the resulting compressed image must be compressed at the next multiple of 8 bits and must be decompressed at the next multiple of 8 bits. Once decompressed, only the number of bits specified in the ISP are to be used for each scan line.

Bottom-to-Top recording algorithm

The Bottom-to-Top recording algorithm formats the image data as a sequence of binary elements that are generated by the unidirectional raster scan operation. The scanning is from left to right (X direction) and from bottom to top (Y direction), as shown in Figure 20. There are no interlaced fields between the parallel scan lines.

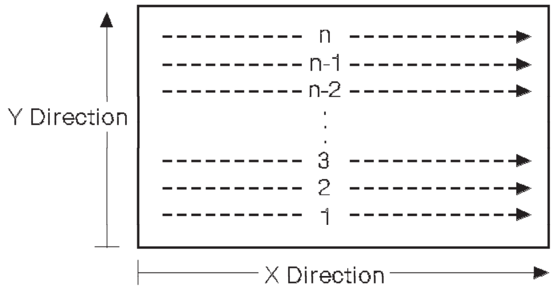


Figure 20. Bottom-to-Top recording algorithm

Each raster scan line is in multiples of 32 bits. If the width of the image is not a multiple of 32, the scan line must be padded with zeros.

Unpadded RIDIC recording algorithm

The Unpadded RIDIC algorithm is identical to the RIDIC recording algorithm except that raster scan lines can be any length; no padding is necessary.

Appendix B. Bilevel, grayscale, and color images

This appendix describes the functions of the image data parameters that represent bilevel, grayscale, and color images.

Related image data parameters

The image data parameters that represent bilevel, grayscale, and color images are:
- IDE Size parameter
- Image LUT-ID parameter
- IDE Structure parameter.

You can represent these images in the following ways:
- Use the IDE Structure parameter to generate a bilevel, grayscale, or color table.
- Specify a nonzero value with the Image LUT-ID parameter to reference an external LUT-ID (LUT) and omit the IDE Structure parameter.
- For bilevel and grayscale images only, use the default value of the Image LUT-ID parameter (LUTID=0) and omit the IDE Structure parameter.

Bilevel images

In all cases below, the IDE size must be 1 (IDESZ=1) in the IDE Size parameter.
- *Using IDE Structure parameter*. A two-index-value bilevel table is generated by the ASFLAG, FORMAT, and SIZE*n* values.
- *With non-zero LUTID*. The LUTID value is used as a pointer to an LUT, which resides in and is controlled by the controlling environment.
- *With LUTID=0*. LUTID=0 means that the standard LUT existing outside the image object is to be used. This LUT, a logical table, has two index values, 0 and 1. It also has entries to give a presentation characteristic to IDE values 0 and 1.

Application note
In AFP environments, the standard LUT defines IDE value of 1 as an significant (toned) pel and IDE value of 0 as an insignificant (untoned) pel.

When both an IDE Structure parameter and a nonzero LUTID value are present, the table generated by the IDE Structure parameter takes precedence over the table identified by the Image LUT-ID parameter.

Grayscale images

- *Using the Image LUT-ID parameter*. A grayscale table can be generated by specifying FORMAT=X'02' (use of the YCrCb model), or FORMAT=X'12' (use of the YCbCr model). In this case, the 2^{SIZE1} value is the number of index value-entry pairs, SIZE2 and SIZE3 can be omitted; if present, they must be zero.

  The IDE Size value must be greater than 1; and each IDE value indexes a gray level in the table. The grayscale table consists of 2^{SIZE1} entries.

  The ASFLAG value determines whether a higher index value is mapped to a brighter or a darker level.

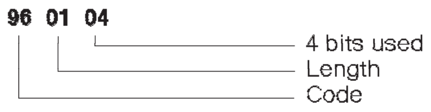
- *With non-zero LUTID*. The LUTID value is used as a pointer to an LUT, which resides in and is controlled by the controlling environment.
- *With LUTID=0*. LUTID=0 means that the standard grayscale table existing outside the image object is to be used.

  The IDE Size value must be greater than 1, and each IDE value indexes a gray level in the table.

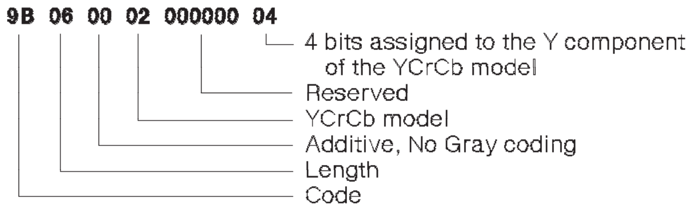
  The grayscale table consists of 2^{IDESZ} entries, where the highest value indexes minimum density and the lowest value indexes maximum density.

When both an IDE Structure parameter and a nonzero LUTID value are present, the table generated by the IDE Structure parameter takes precedence over the table pointed to by the Image LUT-ID parameter.

Example 1: A 16-level grayscale table can be generated by first omitting the Image LUT-ID parameter and then by specifying the IDE Size IDE Size parameter, as follows:



Example 2: The same 16-level grayscale table can be generated with an IDE Structure parameter, as follows:



In this example, the ASFLAG value could be changed to interpret the entry B'1111' to be black.

Color images

There is no default mechanism for expressing color images.

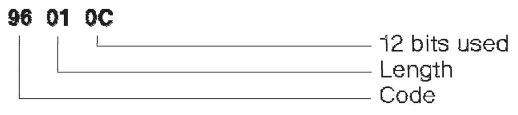
- *Using IDE Structure parameter*. A color table is generated by the ASFLAG, FORMAT, and SIZE*n* values. Each IDE value indexes a color in the table.
- *With non-zero LUTID*. The LUTID value is used as a pointer to an LUT, which resides in and is controlled by the controlling environment.

  Note: Only RGB images can reference an external LUT.

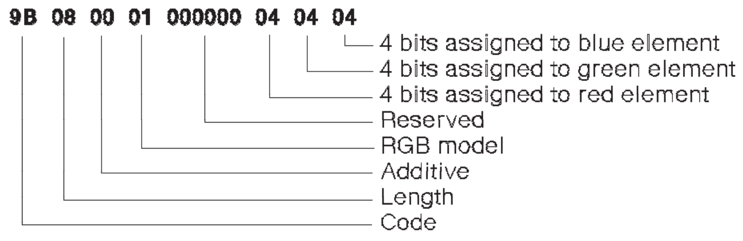
When both an IDE Structure parameter and a nonzero LUTID value are present, the table generated by the IDE Structure parameter takes precedence over the table pointed to by the Image LUT-ID parameter.

Example: For an RGB color image, if red, green, and blue are each represented in 16 levels, the Image Size parameter and IDE Structure parameter are as follows:

- IDE Size parameter:



- IDE Structure parameter:


9B 08 00 01 000000 04 04 04
4 bits assigned to blue element
4 bits assigned to green element
4 bits assigned to red element
Reserved
RGB model
Additive
Length
Code

Refer to the resource appendix in *Mixed Object Document Content Architecture Reference* for a description of the RGB model and the Y component of the YCrCb and YCbCr models.

Appendix C. IOCA Tile Resource

This appendix describes the IOCA Tile Resource. This resource is designed to allow images or image parts that are used multiple times in a same datastream to be downloaded to the receiver only once.

A Tile resource is an IOCA tile, subject to the following rules and conditions:

- A tile resource can contain any parameter otherwise allowed within tiles, except the Include Tile parameter. If a tile resource does contain the Include Tile parameter, exception EC-B811 (Inconsistent Include Tile) exists when the tile is included.
- The content of the Tile Position parameter in the tile resource is ignored. The receiver uses the Tile Position parameter specified in the calling tile instead.
- If both the tile resource and the calling tile contain transparency masks, they are combined using the logical AND operation. A point in the included tile is in the foreground if it is in foreground in both transparency masks. Otherwise, it belongs to the background.
- If only one of the two possible transparency masks is specified, it is used without changes.
- At inclusion time, the tile is treated just as if it were specified locally: the Tile Position parameter in the tile resource is discarded and the transparency mask from the calling tile, if any, is combined with any transparency mask in the included tile. Finally, the included tile, minus the Tile Position and with the possibly changed or added transparency mask is treated as if it appeared instead of the Include Tile parameter in the calling tile.
- Any defaults are applied as if the tile were specified locally.

Table 21 shows the structure of the tile resource.

Table 21. IOCA Tile Resource structure

	X'8C'	Begin Tile parameter		
	X'B5'	Tile Position parameter		
	X'B6'	Tile Size parameter		
[	X'95'	Image Encoding parameter		]
[	X'96'	IDE Size parameter		]
[	X'98'	Band Image parameter		]
[	X'9B'	IDE Structure parameter		]
[	X'9F'	External Algorithm Specification parameter (ignored)		]
[	X'B7'	Tile Set Color parameter		]
[		Transparency Mask		]
[		Image Data or Band Image Data	(S)	]
	X'8D'	End Tile		

Appendix D. MO:DCA environment

This appendix describes how image objects may be included within a MO:DCA-P or MO:DCA-L document for the purpose of interchanging the image objects between a generating node and one or more receiving nodes. Refer to the *Mixed Object Document Content Architecture Reference* for a full description of the MO:DCA data stream and *MO:DCA-L: The OS/2 Presentation Manager Metafile (.met) Format* for a full description of the MO:DCA-L data stream.

IOCA image object in MO:DCA data stream

The Image Data Descriptor (IDD) and Image Picture Data (IPD) structured fields used to carry image objects in MO:DCA-P documents are defined in "Image structured fields in MO:DCA-P data stream" on page 155.

A MO:DCA-L document is a defined interchange set of MO:DCA data stream used for resource documents; that is, documents stored in libraries for later reference. The IDD and IPD structured fields used to carry image objects in MO:DCA-L documents are defined in "Image structured fields in MO:DCA-L data stream" on page 163.

To guarantee interchange, a MO:DCA document carrying an image object must include all information related to the object. The MO:DCA document must therefore contain not only the definition of the image object, but must also provide linkage to the resources the object references.

MO:DCA structured fields are discussed in this appendix only as they relate to IOCA image objects.

Compliance with MO:DCA interchange sets

When image objects are interchanged for the purpose of sending the objects to a display, printer, or any other output device, visual fidelity should be maintained as far as possible. In an attempt to satisfy this objective, IOCA defines the following for the MO:DCA-P and MO:DCA-L environments:

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

In order to comply with a particular MO:DCA interchange set, products that generate image objects must only generate objects that contain image structured fields and values defined in that interchange set. Including structured fields or values not in the interchange set can result in exception conditions being raised by the receiving processor, and exception actions being taken. However, a generator must not rely on a receiver's taking these actions.

In order to conform to a particular MO:DCA interchange set, products that receive image objects and convert them using a processor for output to some device are required to support all the image functions defined in that interchange set.

Image structured fields: general description

This section describes the Image Data Descriptor (IDD) and Image Picture Data (IPD) structured fields. Each structured field consists of a MO:DCA introducer, followed by one or more image control parameters.

Image Data Descriptor (IDD)

The IDD structured field carries the parameters that define the size and resolution of the image presentation space (IPS), and the control parameters required to interpret the image segment.

Structured Field Introducer				
SF Length	X'D3A6FB'	Flags	Sequence Number	Data

Offset	Type	Name	Range	Meaning	M/O
0	CODE	UNITBASE	X'00' – X'01'	Unit base: **X'00'** 10 inches **X'01'** 10 centimeters All other values are reserved.	M
1–2	UBIN	XRESOL	X'0001' – X'7FFF'	Horizontal resolution in image points per unit base	M
3–4	UBIN	YRESOL	X'0001' – X'7FFF'	Vertical resolution in image points per unit base	M
5–6	UBIN	XSIZE	X'0001' – X'7FFF'	Horizontal size of the image presentation space in image points	M
7–8	UBIN	YSIZE	X'0001' – X'7FFF'	Vertical size of the image presentation space in image points	M
9–*n*		SDF		Zero or more self-defining fields	O

The following examples illustrate the relationship between the resolution and size parameters of the IDD and the Image Size parameter.

Example 1: Consider an image with an Image Size parameter that specifies its horizontal and vertical sizes to be two inches, and its horizontal and vertical resolutions to be 300 image points per inch. If one wants the image, when written to the IPS, to remain at two inches in both dimensions, the mandatory parameters of the IDD would have the following values:

UNITBASE	= X'00'
XRESOL	= X'0BB8'
YRESOL	= X'0BB8'
XSIZE	= X'0258'
YSIZE	= X'0258'

Example 2: If one wants the same image to appear at twice the size as the actual image when written to the IPS—that is, the image at the IPS has a horizontal and vertical sizes of four inches—the mandatory parameters of the IDD would have the following values:

UNITBASE	= X'00'
XRESOL	= X'05DC'
YRESOL	= X'05DC'
XSIZE	= X'0258'
YSIZE	= X'0258'

This can be done more easily using the scale-to-fit mapping option in the MO:DCA data stream.

Example 3: Conversely, if one wants the same image to appear at half the size as the actual image when written to the IPS—that is, the image at the IPS has a horizontal and vertical size of one inch—the mandatory parameters of the IDD would have the following values:

UNITBASE	= X'00'
XRESOL	= X'1770'
YRESOL	= X'1770'
XSIZE	= X'0258'
YSIZE	= X'0258'

As can be seen in the previous examples, the horizontal and vertical resolutions of the image as specified in the Image Size parameter are ignored when writing to the IPS. Resolutions specified in the IDD are used instead of the resolutions specified in the Image Size parameter. In the case of an image with undefined resolution, as described in “Image Size” on page 30, each image point in the IOCA image content is mapped to one image point in the IPS. The combination of the horizontal and vertical sizes of the Image Size parameter and the horizontal and vertical resolutions of the IDD determines the actual presentation size of the image.

The image is always written to the IPS starting from the IPS origin. For example, the two-inch square image mentioned in Example 1 appears on the top half of the IPS if the mandatory parameters of the IDD contain the following values:

UNITBASE	= X'00'
XRESOL	= X'0BB8'
YRESOL	= X'0BB8'
XSIZE	= X'0258'
YSIZE	= X'04B0'

If the image cannot fit entirely within the IPS, the IOCA exception condition EC-A902 is raised.

Set Bilevel Image Color

This optional self-defining field specifies a named color value of the significant image points of bilevel images with zero LUT-IDs.

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'F6'	Set Bilevel Image Color	M
1	UBIN	LENGTH	X'04'	Length of the parameters to follow	M

Offset	Type	Name	Range	Meaning	M/O
2	CODE	AREA	X'00'	Applicability area: **X'00'** Foreground All other values are reserved.	M
3			X'00'	Reserved; should be zero	M
4	CODE	NAMECOLR	X'0000' – X'0010', X'FF00' – X'FF08', X'FFFF'	Named colors: **X'0000'** Presentation process default **X'0001'** Blue **X'0002'** Red **X'0003'** Magenta or pink **X'0004'** Green **X'0005'** Cyan or turquoise **X'0006'** Yellow **X'0007'** White **X'0008'** Black **X'0009'** Dark blue **X'000A'** Orange **X'000B'** Purple **X'000C'** Dark green **X'000D'** Dark turquoise **X'000E'** Mustard **X'000F'** Gray **X'0010'** Brown **X'FF00'** Presentation process default **X'FF01'** Blue **X'FF02'** Red **X'FF03'** Magenta or pink **X'FF04'** Green **X'FF05'** Cyan or turquoise **X'FF06'** Yellow **X'FF07'** Presentation process default **X'FF08'** Color of the medium **X'FFFF'** Presentation process default All other values are reserved.	M

If an invalid or unsupported value is encountered in the self-defining field, the entire self-defining field is ignored. If multiple Set Bilevel Image Color self-defining fields appear within the same IDD, the last one encountered is used and all the others are ignored. The IOCA process model should notify the controlling environment when it encounters any of the above exception conditions.

Notes:

1. The medium is typically the physical paper in a printer environment, and the monitor screen in the display environment.
2. The presentation process is typically the program that performs the final imaging step on the medium.
3. This self-defining field is ignored if it is present and the image is not bilevel or with a non-zero LUT-ID.
4. Color specified by X'0007', rendered on presentation devices that do not support white, is device-dependent. For example, some printers simulate white with the color of the medium, which results in white when a white medium is used.

Set Extended Bilevel Image Color

This optional self-defining field specifies a color value and defines the color space and encoding for that value. This SDF is applicable only to significant image points of bilevel images with zero LUT-IDs.

Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'F4'	Set Extended Bilevel Image Color	M
1	UBIN	LENGTH	12–14	Length of the parameters to follow	M
2				Reserved; must be zero	M
3	CODE	ColSpce	X'01', X'04', X'06', X'08', X'40'	Color space: **X'01'** RGB **X'04'** CMYK **X'06'** Highlight color space **X'08'** CIELAB **X'40'** Standard OCA color space	M
4–7				Reserved; must be zero	M
8	UBIN	ColSize1	X'01' – X'08', X'10'	Number of bits in component 1; see color space definitions	M
9	UBIN	ColSize2	X'00' – X'08'	Number of bits in component 2; see color space definitions	M
10	UBIN	ColSize3	X'00' – X'08'	Number of bits in component 3; see color space definitions	M
11	UBIN	ColSize4	X'00' – X'08'	Number of bits in component 4; see color space definitions	M
12–*n*		Color		Color specification; see "Set Extended Bilevel Image Color semantics" for details	M

Set Extended Bilevel Image Color semantics:

ColSpce Is a code that defines the color space and the encoding for the color specification.

Value	**Description**
X'01'	RGB color space. The color value is specified with three components. Components 1, 2, and 3 are unsigned binary numbers that specify the red, green, and blue intensity values, in that order. ColSize1, ColSize2, and ColSize3 are non-zero and define the number of bits used to specify each component. ColSize4 is reserved and should be set to zero. The intensity range for the R,G,B components is 0 to 1, which is mapped to the binary value range 0 to (2^{ColSizeN} – 1), where N=1,2,3. **Architecture note:** The reference white point and the chromaticity coordinates for RGB are defined in SMPTE RP 145-1987, entitled *Color Monitor Colorimetry*, and in RP 37-1969, entitled *Color Temperature for Color Television Studio Monitors*, respectively. The reference white point is commonly known as *Illuminant* D_{6500} or simply *D65*. The R,G,B components are assumed to be gamma-corrected (non-linear) with a gamma of 2.2.
X'04'	CMYK color space. The color value is specified with four components. Components 1, 2, 3, and 4 are unsigned binary numbers that specify the cyan, magenta, yellow, and black intensity values, in that order. ColSize1, ColSize2, ColSize3, and ColSize4 are non-zero and define the number of bits used to specify each component. The intensity range for the C,M,Y,K components is 0 to 1, which is mapped to the binary value range 0 to ($2^{\text{ColSize}N}$ – 1), where *N*=1,2,3,4. This is a device-dependent color space.
X'06'	Highlight color space. This color space defines a request for the presentation device to generate a highlight color. The color value is specified with one to three components. Component 1 is a two-byte unsigned binary number that specifies the highlight color number. The first highlight color is assigned X'0001', the second highlight color is assigned X'0002', and so on. The value X'0000'. specifies the presentation device default color. ColSize1 = X'10' and defines the number of bits used to specify component 1. Component 2 is an optional one-byte unsigned binary number that specifies a percent coverage for the specified color. Percent coverage can be any value from 0% to 100% (X'00'–X'64'). The number of distinct values supported is presentation-device dependent. If the coverage is less than 100%, the remaining coverage is achieved with color of medium. ColSize2 = X'00' or X'08' and defines the number of bits used to specify component 2. A value of X'00' indicates that component 2 is not specified in the color value, in which case the architected default for percent

coverage is 100%. A value of X'08' indicates that component 2 is specified in the color value.

Component 3 is an optional one-byte unsigned binary number that specifies a percent shading, which is a percentage of black that is to be added to the specified color. Percent shading can be any value from 0% to 100% (X'00'–X'64'). The number of distinct values supported is presentation-device dependent. If percent coverage and percent shading are specified, the effective range for percent shading is 0% to (100-coverage)%. If the sum of percent coverage plus percent shading is less than 100%, the remaining coverage is achieved with color of medium. ColSize3 = X'00' or X'08' and defines the number of bits used to specify component 3. A value of X'00' indicates that component 3 is not specified in the color value, in which case the architected default for percent shading is 0%. A value of X'08' indicates that component 3 is specified in the color value.

Implementation note: The percent shading parameter is currently not supported in AFP environments.

ColSize4 is reserved and should be set to zero. This is a device-dependent color space.

Architecture notes:

1. The color that is rendered when a highlight color is specified is device-dependent. For presentation devices that support colors other than black, highlight color values in the range X'0001' to X'FFFF' may be mapped to any color. For bi-level devices, the color may be simulated with a graphic pattern.
2. If the specified highlight color is “presentation device default”, devices whose default color is black use the percent coverage parameter, which is specified in component 2, to render a percent shading.
3. On printing devices, the color of medium is normally white, in which case a coverage of *n*% results in adding (100−*n*)% white to the specified color, or *tinting* the color with (100−*n*)% white. Display devices may assume the color of medium to always be white and use this algorithm to render the specified coverage.
4. The highlight color space can also specify indexed colors when used in conjunction with a Color Mapping Table (CMT) or an Indexed (IX) Color Management Resource (CMR). When used with an Indexed CMR, component 1 specifies a two-byte value that is the index into the CMR and components 2 and 3 are ignored. Note that when both a CMT and Indexed CMRs are used, the CMT is always accessed first. To preserve compatibility with existing highlight color devices, indexed color values X'0000' to X'00FF' are reserved for existing highlight color applications ad devices. That is, indexed color values in range X'0000'

to X'00FF', assuming they are not mapped in a CMT, are mapped directly to highlight colors. Indexed color values in the range X'0100' to X'FFFF', assuming they are not mapped in a CMT, are used to access Indexed CMRs.

X'08' CIELAB color space. The color value is specified with three components. Components 1, 2, and 3 are binary numbers that specify the L, a, b values, in that order, where L is the luminance and a and b are the chrominance differences. Component 1 specifies the L value as an unsigned binary number; components 2 and 3 specify the a and b values as signed binary numbers. ColSize1, ColSize2, and ColSize3 are non-zero and define the number of bits used to specify each component. ColSize4 is reserved and should be set to zero. The range for the L component is 0 to 100, which is mapped to the binary value range 0 to (2^{ColSize1} − 1). The range for the a and b components is −127 to +127, which is mapped to the binary range −($2^{\text{ColSize}N-1}$ − 1) to +($2^{\text{ColSize}N-1}$ − 1).

For color fidelity, 8-bit encoding should be used for each component, that is, ColSize1, ColSize2, and ColSize3 are set to X'08'. When the recommended 8-bit encoding is used for the a and b components, the range is extended to include −128, which is mapped to the value X'80'. If the encoding is less than 8 bits, treatment of the most negative binary endpoint for the a and b components is device-dependent, and tends to be insignificant because of the quantization error.

Architecture note: The reference white point for CIELAB is known as *D50* and is defined in CIE publication 15-2 entitled *Colorimetry*.

X'40' Standard OCA color space. The color value is specified with one component. Component 1 is an unsigned binary number that specifies a named color using a two-byte value from the Standard OCA Color Value Table. ColSize1 = X'10' and defines the number of bits used to specify component 1. ColSize2, ColSize3, ColSize4 are reserved and should be set to zero. This is a device-dependent color space.

All others
Reserved

ColSize1 Defines the number of bits used to specify the first color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary. For example, if ColSize1 = X'06', the first color component has two padding bits.

ColSize2 Defines the number of bits used to specify the second color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

ColSize3 Defines the number of bits used to specify the third color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

ColSize4 Defines the number of bits used to specify the fourth color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

Color Specifies the color value in the defined format and encoding. Note that the number of bytes specified for this parameter depends on the color space. For example, when using 8 bits per component, an RGB color value is specified with 3 bytes, while a CMYK color value is specified with 4 bytes. If extra bytes are specified, they are ignored as long as the self-defining field length is valid.

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

Notes:

1. This self-defining field is ignored if it is present and the image is not bilevel or with a non-zero LUT-ID.
2. This field can coexist with the Set Bilevel Image Color self-defining field.
3. If multiple instances of this field and the Set Bilevel Image Color field are present, the last instance of a supported field is used, while the others are ignored.

If an invalid or unsupported value is encountered in the self-defining field, the entire self-defining field is ignored. The IOCA Process Model should notify the controlling environment if this exception condition appears, or if multiple instances of this field and/or Set Bilevel Image Color field are present.

IOCA Function Set Identification

This optional self-defining field is carried by the IDD described in “Image Data Descriptor (IDD)” on page 146. It specifies the IOCA function set carried by the IPD.

IOCA function sets are defined in “Function sets” on page 91.

Offset	Type	Name	Range	Meaning	M/O
0	CODE	SDFID	X'F7'	IOCA Function Set Identification	M
1	UBIN	LENGTH	X'02'	Length of the parameters to follow	M
2	CODE	CATEGORY	X'01'	Function Set category: **X'01'** Function Set identifier All other values are reserved.	M
3	CODE	FCNSET	X'0A' – X'0B', X'14', X'2A', X'2D'	Function Set identifier: **X'0A'** Function Set 10 **X'0B'** Function Set 11 **X'14'** Function Set 20 **X'28'** Function Set 40 **X'2A'** Function Set 42 **X'2D'** Function Set 45 All other values are reserved.	M

Image Picture Data (IPD)

An IOCA image segment is carried by one or more IPD structured fields.

Structured Field Introducer				
SF Length	X'D3EEFB'	Flags	Sequence Number	image segment

See Chapter 5, “IOCA image segment,” on page 21 for the syntax and description of image segments.

Image structured fields in MO:DCA-P data stream

This section shows the syntax of IDD and IPD in the MO:DCA-P Interchange Set 1 MO:DCA-P IS/1) and Interchange Set 2 (MO:DCA-P IS/2). An IOCA image segment is carried by one or more IPD structured fields.

IDD in MO:DCA-P data stream

Structured Field Introducer				
SF Length	X'D3A6FB'	Flags	Sequence Number	Data

Offset	Type	Name	Range	Meaning	M/O
0	CODE	UNITBASE	X'00'	Unit base: 10 inches	M
1–2	UBIN	XRESOL	X'0001' – X'7FFF'	Horizontal resolution in image points per unit base	M
3–4	UBIN	YRESOL	X'0001' – X'7FFF'	Vertical resolution in image points per unit base	M
5–6	UBIN	XSIZE	X'0001' – X'7FFF'	Horizontal size of the image presentation space in image points (see Note)	M
7–8	UBIN	YSIZE	X'0001' – X'7FFF'	Vertical size of the image presentation space in image points (see Note)	M
9–*n*	CODE	SDF		Set IOCA Function Set Identification self-defining field Set Bilevel Image Color self-defining field Set Extended Bilevel Image Color self-defining field	O

Note: The actual horizontal and vertical sizes of the image presentation space (IPS) can be calculated as follows. They must be equal to or less than 22.754861 inches.

- Horizontal size of the IPS:

  (XSIZE × 10) / XRESOL
- Vertical size of the IPS:

  (YSIZE × 10) / YRESOL

IOCA Function Set Identification

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'F7'	IOCA Function Set Identification	M
1	UBIN	LENGTH	X'02'	Length of the parameters to follow	M

Offset	Type	Name	Range	Meaning	M/O
3	CODE	CATEGORY	X'01'	Function set identifier	M
4	CODE	FCNSET	X'0A' – X'0B', X'2A', X'2D'	Function set: **X'0A'** FS10 for MO:DCA-P IS/1 or MO:DCA-P IS/2 **X'0B'** FS11 for MO:DCA-P IS/2 **X'28'** FS40 **X'2A'** FS42 **X'2D'** FS45	M

Set Bilevel Image Color

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'F6'	Set Bilevel Image Color	M
1	UBIN	LENGTH	X'04'	Length of the parameters to follow	M
2	CODE	AREA	X'00'	Applicability area: **X'00'** Foreground All other values are reserved.	M
3			X'00'	Reserved; should be zero	M

Offset	Type	Name	Range	Meaning	M/O
4–5	CODE	NAMECOLR		Named colors: **X'0000'** Presentation process default **X'0001'** Blue **X'0002'** Red **X'0003'** Magenta or pink **X'0004'** Green **X'0005'** Cyan or turquoise **X'0006'** Yellow **X'0007'** White **X'0008'** Black **X'0009'** Dark blue **X'000A'** Orange **X'000B'** Purple **X'000C'** Dark green **X'000D'** Dark turquoise **X'000E'** Mustard **X'000F'** Gray **X'0010'** Brown **X'FF00'** Presentation process default **X'FF01'** Blue **X'FF02'** Red **X'FF03'** Magenta or pink **X'FF04'** Green **X'FF05'** Cyan or turquoise **X'FF06'** Yellow **X'FF07'** Presentation process default **X'FF08'** Color of the medium **X'FFFF'** Presentation process default All other values are reserved.	M

Note: If the IOCA object is included in the MO:DCA data stream via the Include Object mechanism, and if the relevant IOB structured field contains the Color Specification Triplet X'4E', the OCA color specified in the triplet overrides the OCA color specified in the Set Bilevel Image Color self-defining field.

Set Extended Bilevel Image Color

This optional self-defining field specifies a color value and defines the color space and encoding for that value. This SDF is applicable only to significant image points of bilevel images with zero LUT-IDs.

Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ID	X'F4'	Set Extended Bilevel Image Color	M
1	UBIN	LENGTH	12 - 14	Length of the parameters to follow	M
2				Reserved; must be zero	M

Offset	Type	Name	Range	Meaning	M/O
3	CODE	ColSpce	X'01', X'04', X'06', X'08', X'40'	Color space: **X'01'** RGB **X'04'** CMYK **X'06'** Highlight color space **X'08'** CIELAB **X'40'** Standard OCA color space	M
4–7				Reserved; must be zero	M
8	UBIN	ColSize1	X'01' – X'08', X'10'	Number of bits in component 1; see color space definitions	M
9	UBIN	ColSize2	X'00' – X'08'	Number of bits in component 2; see color space definitions	M
10	UBIN	ColSize3	X'00' – X'08'	Number of bits in component 3; see color space definitions	M
11	UBIN	ColSize4	X'00' – X'08'	Number of bits in component 4; see color space definitions	M
12–*n*		Color		Color specification; see "Set Extended Bilevel Image Color semantics" for details	M

Set Extended Bilevel Image Color semantics:

ColSpce Is a code that defines the color space and the encoding for the color specification.

Value	Description
X'01'	RGB color space. The color value is specified with three components. Components 1, 2, and 3 are unsigned binary numbers that specify the red, green, and blue intensity values, in that order. ColSize1, ColSize2, and ColSize3 are non-zero and define the number of bits used to specify each component. ColSize4 is reserved and should be set to zero. The intensity range for the R,G,B components is 0 to 1, which is mapped to the binary value range 0 to (2^{ColSizeN} – 1), where N=1,2,3.

Architecture note: The reference white point and the chromaticity coordinates for RGB are defined in SMPTE RP 145-1987, entitled *Color Monitor Colorimetry*, and in RP 37-1969, entitled *Color Temperature for Color Television Studio Monitors*, respectively. The reference white point is commonly known as *Illuminant* D_{6500} or simply *D65*. The R,G,B components are assumed to be gamma-corrected (non-linear) with a gamma of 2.2.

X'04' CMYK color space. The color value is specified with four components. Components 1, 2, 3, and 4 are unsigned binary numbers that specify the cyan, magenta, yellow, and black intensity values, in that order. ColSize1, ColSize2, ColSize3, and ColSize4 are non-zero and define the number of bits used to specify each component. The intensity range for the C,M,Y,K components is 0 to 1, which is mapped to the binary value range 0 to ($2^{\text{ColSize}N}$ – 1), where N=1,2,3,4. This is a device-dependent color space.

X'06' Highlight color space. This color space defines a request for the presentation device to generate a highlight color. The color value is specified with one to three components.

Component 1 is a two-byte unsigned binary number that specifies the highlight color number. The first highlight color is assigned X'0001', the second highlight color is assigned X'0002', and so on. The value X'0000'. specifies the presentation device default color. ColSize1 = X'10' and defines the number of bits used to specify component 1.

Component 2 is an optional one-byte unsigned binary number that specifies a percent coverage for the specified color. Percent coverage can be any value from 0% to 100% (X'00'–X'64'). The number of distinct values supported is presentation-device dependent. If the coverage is less than 100%, the remaining coverage is achieved with color of medium. ColSize2 = X'00' or X'08' and defines the number of bits used to specify component 2. A value of X'00' indicates that component 2 is not specified in the color value, in which case the architected default for percent coverage is 100%. A value of X'08' indicates that component 2 is specified in the color value.

Component 3 is an optional one-byte unsigned binary number that specifies a percent shading, which is a percentage of black that is to be added to the specified color. Percent shading can be any value from 0% to 100% (X'00'–X'64'). The number of distinct values supported is presentation-device dependent. If percent coverage and percent shading are specified, the effective range for percent shading is 0% to (100-coverage)%. If the sum of percent coverage plus percent shading is less than 100%, the remaining coverage is achieved with color of medium. ColSize3 = X'00' or X'08' and defines the number of bits used to specify component 3. A value of X'00' indicates that component 3 is not specified in the color value, in which case the architected default for percent shading is 0%. A value of X'08' indicates that component 3 is specified in the color value.

Implementation note: The percent shading parameter is currently not supported in AFP environments.

ColSize4 is reserved and should be set to zero. This is a device-dependent color space.

Architecture notes:

1. The color that is rendered when a highlight color is specified is device-dependent. For presentation devices that support colors other than black, highlight color values in the range X'0001' to X'FFFF' may be mapped to any color. For bi-level devices, the color may be simulated with a graphic pattern.
2. If the specified highlight color is “presentation device default”, devices whose default color is black use the percent coverage parameter, which is specified in component 2, to render a percent shading.
3. On printing devices, the color of medium is normally white, in which case a coverage of *n*% results in adding (100−*n*)% white to the specified color, or *tinting* the color with (100−*n*)% white. Display devices may assume the color of medium to always be white and use this algorithm to render the specified coverage.
4. The highlight color space can also specify indexed colors when used in conjunction with a Color Mapping Table (CMT) or an Indexed (IX) Color Management Resource (CMR). When used with an Indexed CMR, component 1 specifies a two-byte value that is the index into the CMR and components 2 and 3 are ignored. Note that when both a CMT and Indexed CMRs are used, the CMT is always accessed first. To preserve compatibility with existing highlight color devices, indexed color values X'0000' to X'00FF' are reserved for existing highlight color applications ad devices. That is, indexed color values in range X'0000' to X'00FF', assuming they are not mapped in a CMT, are mapped directly to highlight colors. Indexed color values in the range X'0100' to X'FFFF', assuming they are not mapped in a CMT, are used to access Indexed CMRs.

X'08' CIELAB color space. The color value is specified with three components. Components 1, 2, and 3 are binary numbers that specify the L, a, b values, in that order, where L is the luminance and a and b are the chrominance differences. Component 1 specifies the L value as an unsigned binary number; components 2 and 3 specify the a and b values as signed binary numbers. ColSize1, ColSize2, and ColSize3 are non-zero and define the number of bits used to specify each component. ColSize4 is reserved and should be set to zero. The range for the L component is 0 to 100, which is mapped to the binary value range 0 to (2^{ColSize1} − 1). The range for the a and b components is −127 to +127, which is mapped to the binary range −($2^{\text{ColSize}N-1}$ − 1) to +($2^{\text{ColSize}N-1}$ − 1).

For color fidelity, 8-bit encoding should be used for each component, that is, ColSize1, ColSize2, and ColSize3 are set to X'08'. When the recommended 8-bit encoding is used for the a and b components, the range is extended to include −128, which is mapped to the value X'80'. If the encoding is less than 8 bits, treatment of the most negative binary endpoint for the a and b components is device-dependent, and tends to be insignificant because of the quantization error.

Architecture note: The reference white point for CIELAB is known as *D50* and is defined in CIE publication 15-2 entitled *Colorimetry*.

X'40' Standard OCA color space. The color value is specified with one component. Component 1 is an unsigned binary number that specifies a named color using a two-byte value from the Standard OCA Color Value Table. ColSize1 = X'10' and defines the number of bits used to specify component 1. ColSize2, ColSize3, ColSize4 are reserved and should be set to zero. This is a device-dependent color space.

All others
Reserved

ColSize1 Defines the number of bits used to specify the first color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary. For example, if ColSize1 = X'06', the first color component has two padding bits.

ColSize2 Defines the number of bits used to specify the second color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

ColSize3 Defines the number of bits used to specify the third color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

ColSize4 Defines the number of bits used to specify the fourth color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

Color Specifies the color value in the defined format and encoding. Note that the number of bytes specified for this parameter depends on the color space. For example, when using 8 bits per component, an RGB color value is specified with 3 bytes, while a CMYK color value is specified with 4 bytes. If extra bytes are specified, they are ignored as long as the self-defining field length is valid.

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

Notes:

1. This self-defining field is ignored if it is present and the image is not bilevel or with a non-zero LUT-ID.
2. This field can coexist with the Set Bilevel Image Color self-defining field.
3. If multiple instances of this field and the Set Bilevel Image Color field are present, the last instance of a supported field is used, while the others are ignored.

If an invalid or unsupported value is encountered in the self-defining field, the entire self-defining field is ignored. The IOCA Process Model should notify the controlling environment if this exception condition appears, or if multiple instances of this field and/or Set Bilevel Image Color field are present.

IPD in MO:DCA-P data stream

Structured Field Introducer				
SF Length	X'D3EEFB'	Flags	Sequence Number	IOCA Function Set 10 or 11

See "IOCA Function Set 10 (IOCA FS10)" on page 93 and "IOCA Function Set 11 (IOCA FS11)" on page 95 for details.

Notes:

1. An IOCA FS10, FS11, FS42 or FS45 image segment can be split into multiple IPD structured fields. There are no restrictions on how the image segment is split between multiple IPD structured fields. Data beyond the End Segment self-defining field is ignored by receivers.
2. Each image point in IOCA image content is mapped to one image point in the image presentation space.

Image structured fields in MO:DCA-L data stream

This section shows the syntax of IDD and IPD in the MO:DCA-L interchange set. An IOCA image segment is carried by one or more IPD structured fields.

IDD in MO:DCA-L data stream

Structured Field Introducer				
SF Length	X'D3A6FB'	Flags	Sequence Number	Data

Offset	Type	Name	Range	Meaning	M/O
0	CODE	UNITBASE	X'00' – X'01'	Unit base: **X'00'** 10 inches **X'01'** 10 centimeters All other values are reserved.	M
1–2	UBIN	XRESOL	X'0001' – X'7FFF'	Horizontal resolution in image points per unit base	M
3–4	UBIN	YRESOL	X'0001' – X'7FFF'	Vertical resolution in image points per unit base	M
5–6	UBIN	XSIZE	X'0001' – X'7FFF'	Horizontal size of the image presentation space in image points	M
7–8	UBIN	YSIZE	X'0001' – X'7FFF'	Vertical size of the image presentation space in image points	M

IPD in MO:DCA-L data stream

Structured Field Introducer				
SF Length	X'D3EEFB'	Flags	Sequence Number	IOCA Function Set 20

See "IOCA Function Set 20 (IOCA FS20)" on page 102 for details.

Note: An IOCA FS20 image segment can be split into multiple IPD structured fields. Data beyond the End Segment self-defining field is ignored by receivers.

Appendix E. IPDS environment

The Intelligent Printer Data Stream (IPDS) provides the printer subsystem environment for image objects. This appendix describes:
- The context of image objects in the IPDS environment
- IPDS commands specific to images
- Some special considerations when printing an image

For further information about the IPDS architecture, refer to *Intelligent Printer Data Stream Reference*.

IOCA image objects in the IPDS architecture

The IPDS architecture provides various commands to control advanced-function printers. It supports all-points-addressable printing functions that allow text and individual image, graphics, and bar code objects to be positioned and presented at any point on the printed page.

image objects are described to IPDS printers in terms of image segments as defined by IOCA. They are presented in rectangular output areas called *object areas*. These object areas may be positioned at any addressable point on a page, in an overlay or a page segment definition, and may be defined in any one of four orientations (0, 90, 180, and 270) relative to the X-axis of the reference system. The size, position, and orientation of the image object area is defined to the printer by parameters that are specified in the Write Image Control 2 command.

The data within the image presentation space (IPS) can be mapped to the image object area in several different ways, as specified by the Mapping Control Option parameter of the Write Image Control 2 command. These options are as follows:

Scale to Fit
Map the center of the IPS to the center of the object area, and uniformly scale to fit, without changing the aspect ratio of the image. All image data within the IPS is presented when this option is specified.

Center and Trim
Map the center of the IPS to the center of the object area without scaling. Excess image data, if any, is trimmed at object area boundaries.

Position and Trim
Map the upper left corner of the IPS to the object area without scaling, using the specified offset from the image object area origin. Excess image data, if any, is trimmed at object area boundaries.

Replicate and Trim
Map the upper-left corner of the IPS to the object area without scaling, then replicate in both the X and Y directions until the object area is filled. Excess image data, if any, is trimmed at object area boundaries.

Image Point-to-Pel
Map the upper-left corner of the IPS to the origin of the object area. Each image point is mapped to a single output pel: that is, no resolution correction is done. Excess image data, if any, is trimmed at object area boundaries.

Image Point-to-Pel with Double Dot
Same as Image Point-to-Pel, except that each image point is mapped to

four pels in the object area by doubling the image point in both dimensions. No resolution correction is done. Excess image data, if any, is trimmed at object area boundaries.

If the Image Output Control parameters are omitted, the default is Position and Trim.

Note: If the IOCA object is included in a MO:DCA object and the Map Image Object structured field is not present, the MO:DCA default of Scale to Fit applies and the resulting IPDS contains an explicit Scale to Fit Mapping Control Option. For this reason, the IPDS default is very unlikely to be relevant for most applications.

Resolution correction occurs in the Scale to Fit, Center and Trim, Position and Trim and Replicate and Trim mapping options whenever the resolution of the image points in the IPS, in one or both dimensions, is different from the pel resolution of the printer.

Manipulation of image objects can be performed in an IO-Image object state that is entered from any one of three IPDS printer states:
- Page state
- Overlay state
- Page segment state

When the image functions are carried out in the overlay or page segment state, the image data sent to the printer is saved as part of the overlay or page segment definition. It is later included on pages by the Load Copy Control, Include Overlay, or Include Page Segment command.

IPDS IO-Image command set

The IPDS architecture provides the IO-Image command set to convey image information to printers. This command set consists of:
- The Write Image Control 2 command, which defines where and how to present an image object
- The Write Image 2 command, which contains an image segment

Write Image Control 2

The Write Image Control 2 command is identified by command code X'D63E', and is sent to the printer before the Write Image 2 command. It tells the printer that the Write Image 2 commands that follow are directed to an image object area on the current page, overlay, or page segment.

This command defines the size, placement, and orientation of the image object area. It also establishes the parameters required to interpret the image segment.

The Write Image Control 2 data is made up of the following three consecutive self-defining fields:

Image Area Position (IAP)
This mandatory self-defining field defines the position and orientation of the image object area relative to a reference coordinate system.

Image Output Control (IOC)
This optional self-defining field specifies the size of the image object area and mapping option for mapping the image presentation space to the image object area.

If it is omitted, the Position and Trim mapping option applies where the offsets are zero, and the image object area size is the same as the image presentation space size as defined in the IDD self-defining field.

Image Data Descriptor (IDD)
This mandatory self-defining field specifies the parameters that define the image presentation space size, and control parameters required to interpret the image segment.

Refer to the *Intelligent Printer Data Stream Reference* for a complete description of the above self-defining fields.

Write Image 2

The Write Image 2 command is identified by command code X'D64E'. One or more Write Image 2 commands carry one IOCA image segment to the printer.

All image segments are executed in Immediate mode. That is, they are not retained or stored as named segments, but processed immediately when the printer receives them.

There are no quantity restrictions on data sent to the printer in a single Write Image 2 command, except for the 32K-length limit of the command. An image segment, delimited by the Begin Segment and End Segment self-defining fields, may span two or more consecutive Write Image 2 commands.

The IO-Image Command Set allows for image segments that conform to Function Set 10 (IOCA FS10). See Chapter 7, “Compliance,” on page 91.

Some IPDS printers also support:

- RL4, ABIC, and G3 MR—Modified READ compression algorithms
- Bit ordering
- Grayscale images that use the default standard LUT-ID (LUTID=0) and have IDE size values greater than one

Exception handling

A data-stream exception occurs when the printer detects an invalid or unsupported command, control, or parameter value in the data stream received from the controlling environment. The IPDS architecture assigns a unique exception code to each exception condition.

The IPDS architecture defines exception conditions and actions that may be detected in IOCA image segments carried in the IPDS data stream. They are compatible with IOCA-defined exception conditions and actions.

The IPDS Exception Identifier consists of the two-byte EC identifier defined by IOCA, prefixed by an IPDS exception class value of X'05'. The exception class value is used to distinguish between the two-byte EC identifiers assigned by IOCA, and other two-byte EC identifiers assigned to presentation text (PTOCA), graphics (GOCA), and bar code (BCOCA) objects.

Unsupported IOCA functions in an IPDS environment

Not all IOCA printers support the full range of IOCA function; these printers return an appropriate NACK if unsupported IOCA self-defining fields or values are included in an image. For example, if an IOCA FS11, FS42, or FS45 image is sent to an IPDS printer that only supports IOCA FS10, the printer will encounter a data stream error and will return one or more of the following exception conditions, depending on which error is encountered first.

Table 22. Exception conditions for unsupported IOCA functions

IOCA exception condition	IPDS exception identifier	Error encountered
EC-0001	X'0500..01'	Invalid or unsupported self-defining fields: Band Image Data Band Image parameter Begin Tile parameter Begin Transparency Mask parameter Color Palette parameter End Tile parameter End Transparency Mask parameter External Algorithm Specification parameter IDE Structure parameter Image Data Image Lookup Table ID parameter Image Size parameter Image Subsampling parameter Include Tile parameter Tile Position parameter Tile Set Color parameter Tile Size parameter Tile TOC parameter
EC-0003	X'0500..03'	Image Encoding parameter length error
EC-9510	X'0595..10'	Unsupported compression algorithm
EC-9610	X'0596..10'	Too many bits per IDE
EC-9710	X'0597..10'	Unsupported Look-Up Table identifier

Additional related commands

The following commands are used for query and resource management functions. Only an overview of these commands is presented here. They 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 supported command sets. The information requested is returned in the Special Data Area of the Acknowledge Reply. The command sets and the 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 printable area currently available, symbol-set support, image and coded-font resolution, and color support.

Special notes

This section describes special considerations for the IPDS environment.

Image segment in IO-Image command set

For untiled image contents, the image size is specified in the Image Size parameter which is a mandatory parameter within an untiled image content. An exception condition occurs if the parameter either is not found, appears more than once, appears before the Begin Image Content, or appears after the first image data self-defining field. In this situation, the IOCA standard exception action and IPDS Alternate Exception Action (AEA) is to process the rest of the image segment.

Since the Image Size parameter is mandatory in each untiled image content, its contents (except for values in Unit Base, Horizontal, and Vertical Resolutions) must be checked for validation. Exceptions occur under the following conditions:

- The Image Size parameter specifies an unknown horizontal image size (HSIZE=0), and an image compression algorithm other than InfoPrint MMR—Modified Modified Read is selected in the Image Encoding parameter. The IOCA exception action and the IPDS AEA is to skip to the end of the image segment.
- The size detected from the image data is different from that specified in the Image Size parameter. The IOCA exception action and the IPDS AEA is to use the size of the image detected from the image data.

  When the image size extends beyond the XSIZE or YSIZE of the image presentation space, an exception condition occurs. The IOCA exception action and the IPDS AEA is to write only portions of the image that are within the image presentation space, and discard all portions that extend outside it. The portions that are not written onto are filled with zeros.

Each image point in IOCA image content is mapped to one image point in the image presentation space. The relationship between the resolution and size parameters of the IDD and the Image Size parameter are further described in "Image Data Descriptor (IDD)" on page 146.

Interpretation of IDE value

Bilevel images are represented by an IDE size of one. Each IDE can represent two values, B'1' or B'0'. In the IPDS architecture, an IDE value of B'1' represents a significant bit that is an image point representing a toned pel in the printer, while B'0' represents an insignificant bit that is an image point representing an untoned pel in the printer.

Image presentation space mapping

The image to be printed is represented as an array of image points in the image presentation space after execution of the image segment. The size of the image presentation space and the resolution of the image points within it are defined in the IPDS WIC2 IDD self-defining field.

The size of the Image object area is defined in the IPDS WIC2 IOC self-defining field.

Printing the image data requires the printer to map the *logical* image existing in the image presentation space to a *physical* image in the image object area on the page.

Special notes

The mapping options specified in the IPDS WIC2 IOC self-defining field define how the image will be located with respect to the object area, and whether scaling is needed.

Resolution correction occurs in the Scale to Fit, Center and Trim, Position and Trim and Replicate and Trim mapping options whenever the resolution of the image points in the IPS, in one or both dimensions, is different from the pel resolution of the printer.

Appendix F. Notes for IOCA generators

IOCA is designed to support printing of images at high speed. However, it is relatively easy to construct syntactically valid images that have extremely poor performance. This is particularly true in printing color, since high speed color printing is a very demanding task.

This appendix reviews some of the most important concerns that should be addressed to ensure that the images print with both high performance and good quality.

General considerations

When printing images, the concern is the complexity of an *average* page. The printer control unit in fast continuous forms printers generally processes pages in advance of printing. Thus, a sequence of several "easy" pages, followed by a "hard" page may print at rated speed, since the average page complexity is still acceptable.

An letter-sized page at 600 dpi contains roughly 33 million pixels. Image operations on images of such a size, even when they are black and white bilevel (one bit per pel), tend to be prohibitively expensive. If at all possible, the images should be generated at the right size and orientation.

Generators should bear in mind that the in MO:DCA data streams the default mapping option is Scale to Fit. A Map Image Object specifying Position and Trim should be specified explicitly. If the image presentation space dimensions do not quite match the image object dimensions specified in the Object Area Descriptor, the default mapping forces the printer to scale the image. Even if the scaling is unnoticeable (for example, there is a difference of one scan line between the object and image lengths), it extracts a significant performance penalty. In contrast, trimming or padding of bilevel images can usually be performed at rated speed.

Unlike scaling, rotation of images can sometimes be performed at high speed. For color images, this subject is discussed below. For black and white bilevel images, some printers can perform rotation in runend domain. In the runend domain, only the transitions between black and white runs in the image are recorded. For images containing text or line art, there are few runs per scan line and the runend domain algorithms perform very efficiently. Halftone images, on the other hand, are far less suitable for such an approach.

Given the complexity of the rotation issue, it is much better to generate images at the proper orientation. Note that in continuous forms printers, the default orientation for one-up printing is landscape. To achieve high image performance in this context, the images should be pre-rotated 90 degrees and should have the rotation in the Object Area Position set to 270 degrees. In most printers, assuming that one-up prints at 90-degree landscape orientation, this avoids rotation in the printer control unit.

Printing halftones poses several distinct challenges:

- *Compressed image size*. High frequency halftones tend to compress very poorly. For example, 212lpi halftones used in some of the color printers cause the G4

MMR compression to actually expand data. If the halftoned area is not large, or if the image is light, this is not a particular concern. If the halftoned images are causing performance difficulties, lower frequency screens of 106 lpi or below should be used.

- *Device dependency*. Halftoned images are device dependent. The halftone screens are built for a particular type of the print engine. Moreover, each print engine behaves differently and behavior changes unpredictably with time, based on many environmental and internal factors. For the best quality, the halftones should be calibrated frequently, using tools such as the Halftone Management System that is part of Infoprint® Manager for AIX®. If quality output is desired, halftone images should not be archived. The generators should rather archive the original color or grayscale and generate the halftoned IOCA when the print device characteristics are known. Black and white text and linework are not device-specific and can be archived safely.
- *Scaling impact*. Scaling halftoned images by non-integer factors results in artifacts and unacceptable output quality. The generators should ensure that the image is generated with the same resolution used by the printer. The only exception is if the printer resolution is an even multiple of the image resolution. For example, printing a 300-dpi image on a 600-dpi printer produces a good quality image, albeit at 300 dpi. Printing a 240-dpi image on a 600-dpi printer results in visible artifacts and poor quality, because 600 is not evenly divisible by 240.

Most bilevel IOCA images are generated using the RIDIC recording algorithm and G4 compression algorithm. The generators should keep in mind that RIDIC requires the image scan lines to be padded to a multiple of eight before compression. Note that TIFF images, which are often used as a source for generating IOCA, also support G4, but do not require that the scan lines be padded. Rewrapping G4 TIFF images with widths that are not multiples of 8 in IOCA is a major source of errors.

If a TIFF image has a width that is not a multiple of 8, the generators should decompress the image, pad each scanline to a multiple of 8 and then recompress. Alternatively, the generators should use the Unpadded RIDIC recording algorithm, which does not require that the scanlines be padded. Be warned, however, that not all printers support the Unpadded RIDIC recording algorithm.

Function Set 42 considerations

Function Set 42 should be used only for those color printers that do not support the full color Function Set 45. Images using one bit per spot (per pel per color component) have worse quality than images using 8 bits per spot. The greater bit depth also allows a range of more sophisticated compression schemes, so the full color images also require less data per unit of image area than images using one bit per spot.

Since the images in Function Set 42 have one bit per spot, the colors are obtained by halftoning. This includes any color used in the image except fully saturated cyan, magenta, yellow, black, or white.

Color printers supporting FS42 will likely use high frequency screens. Function Set 42 also supports the ABIC compression algorithm, which does compress the halftoned data. ABIC, however, tends to be very expensive to decompress. In some cases, it may be preferable to send FS42 images uncompressed.

Tiles compressed using the Solid Fill Rectangle algorithm are not affected by scaling, regardless of the color specified in the related Tile Set Color or Set Bilevel Image Color.

Images that contain just black or other fully saturated color text and line work can also be scaled in the printer without excessive loss of quality, though performance still suffers.

Function Set 42 images containing CMYK tiles cannot be transformed to print on a bilevel (black and white) printer with reasonable performance and quality. These images are halftoned, which involves an information loss. To obtain a bilevel image, the CMYK bilevel image must first be analyzed and transformed back into 8-bits/band CMYK. The 8-bit data can then be used to compute the 8-bit luminance (grayscale), which in turn has to be halftoned for the bilevel output device. The process is very compute-intensive and, given the information loss at several stages, likely to lead to poor-quality output. If the application anticipates having to present the image on different devices, the full color image, either 8 bit CMYK or, even better, a device-independent format like CIELab, should be archived. Applications are strongly discouraged from trying to recover device-independent color from the 1-bit/band CMYK. Since each output CMYK device has different characteristics, even printing a CMYK image halftoned for one device on a different device might lead to poor quality.

Function Set 45 considerations

To achieve good performance and quality with the full color images, it is crucial that the images are compressed using a compression algorithm that is best matched to the type of the image:

- InfoPrint MMR—Modified Modified READ algorithm is obsolete. Using G4 MMR compression almost always results in better compression.
- MMR algorithms are well-suited for compressing the text and line art. If the image contains halftones, the compression ratios degrade as the screens get finer. At roughly 150 lines per inch, the G4 algorithm generally compresses the data. At 212 lines per inch (high end color printers tend to use frequencies of 212 lpi and above), the MMR algorithms cause the image to actually expand, possibly by a factor of two or more. For such images, using no compression is currently the best choice.
- The ABIC compression algorithm compresses even high frequency halftones. ABIC is a complex algorithm and decompressors may be slow, depending on the printer. In some cases, an image compressed with ABIC takes longer to download and decompress than the same image uncompressed, even though the uncompressed image has more than twice the amount of data. The performance of the ABIC decompressor in the printer should be tested before the decision is made to use ABIC.
- JPEG algorithm is well-suited for compressing continuous tone images such as photographs. Using JPEG on text, line art, pie charts and similar images results in artifacts and unacceptable image quality. Such images should be compressed using the TIFF LZW algorithm.
- TIFF LZW algorithm is an excellent general-purpose lossless algorithm. It is particularly well suited to compressing large areas of uniform color. While the output very rarely expands (unlike MMR-type algorithms on halftones), it generally achieves only 10% compression on the continuous tone images. For such images, JPEG should be used.

FS45 considerations

Using a valid compression algorithm that is poorly matched to the data does not cause any exception to be raised, but negatively affects either the printer performance or output quality or both.

Given the large datasets needed to print full color images, it is even more crucial that the images be generated at the right size, resolution, and orientation.

Notices

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

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

References in this document to InfoPrint Solutions Company products, product features, programs or services do not imply that InfoPrint Solutions Company intends to make such products, product features, programs or services available in all countries in which InfoPrint Solutions Company operates or does business.

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

InfoPrint Solutions Company, LLC
6300 Diagonal Hwy 002J
Boulder, CO 80301-9270
U.S.A.

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

InfoPrint Solutions Company, LLC
6300 Diagonal Hwy 002J
Boulder, CO 80301-9270
U.S.A.

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

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

Trademarks

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

- Advanced Function Presentation
- Advanced Function Printing
- AFP
- Bar Code Object Content Architecture
- BCOCA
- Color Management Object Content Architecture
- InfoPrint
- Infoprint
- Intelligent Printer Data Stream
- IPDS
- Mixed Object Document Content Architecture
- MO:DCA
- Print Services Facility
- Ricoh

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

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.

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

UNIX is a registered trademark of The Open Group in the United States and other countries.

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

Glossary

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.

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*.

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

Advanced Function Presentation (AFP). The InfoPrint Solutions Company strategic environment for presentation.

AEA. See *alternate exception action*.

AFP. See *Advanced Function Presentation*.

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

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

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*.

ANSI. See *American National Standards Institute*.

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

aspect ratio. The ratio of the horizontal size of a picture to the vertical size of the picture.

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

B

background. The part of a presentation space that is not occupied with object data. Contrast with *foreground*.

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

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

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.

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

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

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

C

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

character. A member of a set of elements used for the organization, control, or representation of data. A character can be either a graphic character or a control character. In bar codes, a single group of bars and spaces that represent an individual number, letter, punctuation mark, or other symbol.

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

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

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

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

color model. The model by which a color is specified. For example, the RGB model specifies color in terms of three intensities for red (R), green (G), and blue (B).

color of medium. The color of a presentation space before any data is added to it. Synonymous with *reset color*.

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

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

command set. A collection of IPDS commands.

compression algorithm. An algorithm used to compress image data. Compression of image data can decrease the volume of data required to represent an image.

construct. An architected set of data such as a structured field or a triplet.

continuous-form media. Connected sheets. An example of connected sheets is sheets of paper connected by a perforated tear strip. Contrast with *cut-sheet media*.

controlling environment. The environment in which an object is embedded, for example, the IPDS and MO:DCA data streams.

coordinate system. A Cartesian coordinate system. An example is the image coordinate system that uses the fourth quadrant with positive values for the Y axis. The origin is the upper left-hand corner of the fourth quadrant. A pair of (x,y) values corresponds to one image point. Each image point is described by an image data element.

coordinates. A pair of values that specify a position in a coordinate space. See also *absolute coordinate* and *relative coordinate*.

cut-sheet media. Unconnected sheets. Contrast with *continuous-form media*.

D

data block. In the IPDS architecture, a rectangular area in a presentation space into which a data object is mapped. The presentation space can be for a page or an overlay. Examples are a graphics block, an image block, and a bar code block. Synonymous with *object area*.

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

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*.

default. A value, attribute, or option that is assumed when none has been specified and one is needed to continue processing.

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.

digital half-toning. A method used to simulate gray levels on a bilevel device.

digital image. An image whose image data was sampled at regular intervals to produce a digital representation of the image. The digital representation is usually restricted to a specified set of values.

document content architecture. A family of architectures that define the syntax and semantics of the document component. See also *structured field*.

E

EBCDIC. See *Extended Binary-Coded Decimal Interchange Code*.

exception. One of the following:

- An invalid or unsupported data-stream construct
- In the IPDS architecture, a condition requiring host notification
- 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*.

Extended Binary-Coded Decimal Interchange Code (EBCDIC). A coded character set that consists of eight-bit coded characters.

F

foreground. The part of a presentation space that is occupied with object data. 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*.

format. The arrangement or layout of data on a physical medium or in a presentation space.

function set. A collection of architecture constructs and associated values. Function sets can be defined across or within subsets.

G

Graphics Object Content Architecture (GOCA). An architected collection of constructs used to interchange and present graphics data.

grayscale image. An 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.

H

hexadecimal. A number system with a base of sixteen. The decimal digits 0 through 9 and characters A through F are used to represent hexadecimal digits. The hexadecimal digits A through F correspond to the decimal numbers 10 through 15, respectively. An example of a hexadecimal number is X'1B', which is equal to the decimal number 27.

host. In the IPDS architecture, a computer that drives a printer. In IOCA, the host is the controlling environment.

I

IDE. See *image data element*.

IEEE. Institute of Electrical and Electronics Engineers.

IDP. See *image data parameter*.

image. An electronic representation of a picture produced by means of sensing light, sound, electron radiation, or other emanations coming from the picture or reflected by the picture. An image can also be generated directly by software without reference to an existing picture.

image block. A rectangular area on a logical page into which an image presentation space is mapped.

image content. Image data and its associated image data parameters.

image coordinate system. An X,Y Cartesian coordinate system using only the fourth quadrant with positive values for the Y axis. The origin of an image coordinate system is its upper left hand corner. An X,Y coordinate specifies a presentation position that corresponds to one and only one image data element in the image content.

image data. Rectangular arrays of raster information that define an image.

image data element (IDE). A basic unit of image information. An image data element expresses the intensity of a signal at a corresponding image point. An image data element can use a look-up table to introduce a level of indirection into the expression of grayscale or color.

image data parameter (IDP). A parameter that describes characteristics of image data.

image distortion. Deformation of an image such that the original proportions of the image are changed and the original balance and symmetry of the image are lost.

image object. An object that contains image data. See also *object*.

Image Object Content Architecture (IOCA). An architected collection of constructs used to interchange and present images.

image point. A discrete X,Y coordinate in the image presentation space. See also *addressable position*.

image presentation space (IPS). A two-dimensional conceptual space in which an image is generated.

image segment. Image content bracketed by Begin Segment and End Segment self-defining fields. See also *segment*.

IM image. A migration image object that is resolution dependent, bilevel, and cannot be compressed or scaled. Contrast with *IO image*.

IM-image command set. In the IPDS architecture, a collection of commands used to present IM-image data in a page, page segment, or overlay.

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*.

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.

IOCA. See *Image Object Content Architecture*.

IO image. An image object containing IOCA constructs. Contrast with *IM image*.

IO-image command set. In the IPDS architecture, a collection of commands used to present IOCA data in a page, page segment, or overlay.

IPDS. See *Intelligent Printer Data Stream*.

IPS. See *image presentation space*.

ISO. See *International Organization for Standardization*.

L

local identifier (LID). An identifier that is mapped by the environment to a named resource.

location. A site within a data stream. A location is specified in terms of an offset in the number of structured fields from the beginning of a data stream, or in the number of bytes from another location within the data stream.

logical unit. A unit of linear measurement expressed with a unit base and units per unit-base value. For example, in MO:DCA and IPDS architectures, the following logical units are used:

- 1 logical unit = 1/1440 inch (unit base = 10 inches, units per unit base = 14400)
- 1 logical unit = 1/240 inch (unit base = 10 inches, units per unit base = 2400)

Synonymous with *L-unit*.

look-up table (LUT). A logical list of colors or intensities. The list has a name and can be referenced to select a color or intensity. See also *color table*.

lossless. A form of image transformation in which all of the data is retained. Contrast with *lossy*.

lossy. A form of image transformation in which some of the data is lost. Contrast with *lossless*.

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 = 14400)

- 1 L-unit = 1/240 inch (unit base = 10 inches, units per unit base = 2400)

Synonymous with *logical unit*.

LUT. See *look-up table*.

M

meaning. A table heading for architecture syntax. The entries under this heading convey the meaning or purpose of a construct. A meaning entry can be a long name, a description, or a brief statement of function.

measurement base. A base unit of measure from which other units of measure are derived.

media. Plural of medium. See also *medium*.

medium. A two-dimensional conceptual space with a base coordinate system from which all other coordinate systems are either directly or indirectly derived. A medium is mapped onto a physical medium in a device-dependent manner.

mil. 1/1000 inch.

Mixed Object Document Content Architecture (MO:DCA). An architected, device-independent data stream for interchanging documents.

N

NACK. See *Negative Acknowledge Reply*.

name. A table heading for architecture syntax. The entries under this heading are short names that give a general indication of the contents of the construct.

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 resource. A resource that invokes nested resources. See also *nested resource*.

neutral white. A color attribute that gives a device-dependent default color, typically white on a screen and black on a printer.

nonprocess runout (NPRO). An operation that moves sheets of physical media through the printer without printing on them. This operation is used to stack the last printed sheet.

no operation (NOP). A construct whose execution causes a product to proceed to the next instruction to be processed without taking any other action.

N-up. The presentation of a fixed number of pages on a side of a physical medium. For example, 4-up is the presentation of four pages on a side.

O

object. A collection of structured fields. The first structured field provides a begin-object function, and the last structured field provides an end-object function. The object can contain one or more other structured fields whose content consists of one or more data elements of a particular data type. An object can be assigned a name, which can be used to reference the object. Examples of objects are text, font, graphics, image, and formatted data objects. Something that a user works with to perform a task.

object area. In MO:DCA, a rectangular area in a presentation space into which a data object is mapped. The presentation space can be for a page or an overlay. Examples are a graphics object area, an image object area, and a bar code object area. Synonymous with *data block*.

object data. A collection of related data elements bundled together. Examples of object data include graphic characters, image data elements, and drawing orders.

offset. A table heading for architecture syntax. The entries under this heading indicate the numeric displacement into a construct. The offset is measured in bytes and starts with byte zero. Individual bits can be expressed as displacements within bytes.

orientation. The angular distance a presentation space or data block is rotated in a specified coordinate system, expressed in degrees and minutes. For example, the orientation of printing on a physical medium, relative to the X_m axis of the X_m,Y_m coordinate system. See also *presentation space orientation*.

origin. The point in a coordinate system where the axes intersect. Examples of origins are the addressable position in an X_m,Y_m coordinate system where both coordinate values are zero.

orthogonal. Intersecting at right angles. An example of an orthagonal relationship is the positional relationship between the axes of a Cartesian coordinate system.

P

page. A data stream object delimited by a Begin Page structured field and an End Page structured field. A page can contain text, image, graphics, and bar code data. In the IPDS architecture, a page can be copied a specified number of times with or without modification. The final representation of such an object on a physical medium.

physical medium. A physical entity on which information is presented. Examples of a physical medium are a sheet of paper and a display screen. See also *medium*.

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. Picture elements per inch is often used as a measurement of presentation granularity. Synonymous with *pixel* and *picture element*.

physical printable area. A bounded area defined on the physical medium 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.

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. A unit of measure used mainly for measuring typographical material. There are seventy-two points to an inch. In GOCA, a parameter that specifies the position within the drawing order coordinate space.

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

Positive Acknowledge Reply (ACK). In the IPDS architecture, a reply to an IPDS command that has its ARQ flag on and in which no exception is reported. Contrast with *Negative Acknowledge Reply*.

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 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 *image 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*.

Presentation Text Object Content Architecture (PTOCA). An architected collection of constructs used to interchange and present presentation text data.

R

range. A table heading for architecture syntax. The entries under this heading give numeric ranges applicable to a construct. The ranges can be expressed in binary, decimal, or hexadecimal. The range can consist of a single value.

raster pattern. A rectangular array of pels arranged in rows called scan lines.

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

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*.

repeating group. A group of parameter specifications that can be repeated.

reserved. Having no assigned meaning and put aside for future use. The content of reserved fields is not used by receivers, and should be set by generators to a specified value, if given, or to binary zeros. A reserved field or value can be assigned a meaning by an architecture at any time.

reset color. The color of a presentation space before any data is added to it. Synonymous with *color of medium*.

resolution. 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. The number of addressable pels per unit of length.

resolution correction. A method used to present an image on a printer without changing the physical size or proportions of the image when the resolutions of the printer and the image are different.

resolution-correction ratio. The ratio of a printer's physical resolution to an image presentation space's resolution.

resolution modification. A method used to write an image on an image presentation space without changing the physical size of the image when the resolutions of the presentation space and the image are different.

resource. An object that is referenced by a data stream or by another object to provide data or information. Resource objects can be stored in libraries. In MO:DCA, resource objects can be contained within a resource group. Examples of resources are fonts, overlays, and page segments.

retired. Set aside for a particular purpose, and not available for any other purpose. Retired fields and values are specified for compatibility with existing products and identify one of the following:

- Fields or values that have been used by a product in a manner not compliant with the architected definition
- Fields or values that have been removed from an architecture

rotation. The orientation of a presentation space with respect to the coordinate system of a containing presentation space. Rotation is measured in degrees in a clockwise direction. Zero-degree rotation exists when the angle between a presentation space's positive X axis and the containing presentation space's positive X axis is zero degrees.

row. A subarray that consists of all elements that have an identical position within the high dimension of a regular two-dimensional array.

S

SBIN. A data type for architecture syntax, that indicates that one or more bytes be interpreted as a signed binary number, with the sign bit in the high-order position of the leftmost byte. Positive numbers are represented in true binary notation with the sign bit set to B'0'. Negative numbers are represented in twos-complement binary notation with a B'1' in the sign-bit position.

scaling. Making all or part of a picture smaller or larger by multiplying the coordinate values of the picture by a constant amount. If the same multiplier is applied along both dimensions, the scaling is uniform, and the proportions of the picture are unaffected. Otherwise, the scaling is anamorphic, and the proportions of the picture are changed. See also *anamorphic scaling*.

scaling ratio. The ratio of an image-block size to an image-presentation-space size.

scan line. A series of picture elements. Scan lines in raster patterns form images. See also *picture element* and *raster pattern*.

segment. In IOCA, image content bracketed by Begin Segment and End Segment self-defining fields. See also *image segment*.

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.

semantics. The meaning of the parameters of a construct. See also *syntax*.

standard action. The architecture-defined action to be taken on detecting an exception condition, when the environment specifies that processing should continue.

structured field. A self-identifying, variable-length, bounded record, which can have a content portion that provides control information, data, or both.

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.

synchronous exception. In the IPDS architecture, a data-stream or resource-storage exception that must be reported to the host before a printer can return a Positive Acknowledge Reply or can increment the received-page counter for a page containing the exception. Synchronous exceptions are those with action code X'01', X'0C', or X'1F'. See also *data-stream exception*. Contrast with *asynchronous exception*.

syntax. The rules governing the structure of a construct.

T

toned. Containing marking agents such as toner or ink. Contrast with *untoned*.

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 presentation space. Synonymous with *clipping*.

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.

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.

U

UBIN. A data type for architecture syntax, indicating one or more bytes to be interpreted as an unsigned binary number.

unarchitected. Identifies data that is neither defined nor controlled by an architecture. Contrast with *architected*.

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

unit base. A one-byte code that represents the length of the measurement base. For example, X'00' might specify that the measurement base is ten inches.

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

V

valid printable area (VPA). The intersection of a logical page with the area of the medium presentation space in which printing is allowed. If the logical page is a secure overlay, the area in which printing is allowed is the physical printable area. If the logical page is not a secure overlay and if a user printable area is defined, the area in which printing is allowed is the intersection of the physical printable area with the user printable area. If a user printable area is not defined, the area in which printing is allowed is the physical printable area.

Related publications

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

Architecture publications

- *Bar Code Object Content Architecture Reference*, S544-3766
- *Color Management Object Content Architecture™ Reference*, S550-0511
- *Font Object Content Architecture Reference*, S544-3285
- *Intelligent Printer Data Stream Reference*, S544-3417
- *Graphics Object Content Architecture Reference*, SC31-6804
- *Graphics Object Content Architecture for AFP Reference*, S544-5498
- *Mixed Object Document Content Architecture Reference*, SC31-6802
- *MO:DCA-L: The OS/2 Presentation Manager Metafile (.met) Format*, S550-1135-00
- *Presentation Text Object Content Architecture Reference*, SC31-6803

Advanced Function Presentation publications

- *Advanced Function Presentation: Printer Information*, G544-3290
- *Advanced Function Presentation: Printer Summary*, G544-3135
- *Advanced Function Printing™: Host Font Data Stream Reference*, S544-3289
- *AFP Application Programming Interface: Programming Guide and Reference*, S544-3872
- *AFP Conversion and Indexing Facility: Application Programming Guide*, G544-3824
- *AFP Fonts: Font Summary*, G544-3810
- *AFP Fonts: Technical Reference for Code Pages*, S544-3802
- *AFP Workbench for Windows® 95 and Windows NT®: Technical Reference*, S544-5602
- *Guide to Advanced Function Presentation*, G544-3876
- *Page Printer Formatting Aid: User's Guide and Reference*, S544-5284

Further reading

These publications describe image compression algorithms:

- Abramson, Norman. *Information Theory and Coding*. New York: McGraw-Hill, 1963.
- Arps, R., T. Truong, D. Lu, R. Pasco, and T. Friedman, "A multipurpose VLSI chip for adaptive data compression of bilevel images". *IBM Journal of Research and Development*, Volume 32, No. 6 (November 1988).
- "Binary-image-manipulation Algorithms in Image View Facility". *IBM Journal of Research and Development*, vol. 31, no. 1 (January 1987).
- International Organization for Standardization and International Electrotechnical Commission. ISO/IEC International Standard 10918-1. 1994.
- *Composed Page Data Stream Architecture IS & TG Architecture Memorandum*. AR-7262-03-POK. Poughkeepsie, NY: IBM®.
- International Telecommunications Union-Telecommunication Standardization Sector. *Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus*. Terminal Equipment and Protocols for Telematic Services Recommendations of the T Series, Recommendation T.6. ITU–TSS Volume VII, Fascicle VII.3:.
- ________. *Standardization of Group 3 Facsimile Apparatus for Document Transmission*. Terminal Equipment and Protocols for Telematic Services Recommendations of the T Series, Recommendation T.4. ITU–TSS Volume VII, Fascicle VII.3.
- ________. Terminal Equipment and Protocols for Telematic Services Recommendations of the T Series, Recommendation T.81. 1993.
- Pennebaker, William B., and Joan L. Mitchell. *JPEG: Still Image Data Compression Standard*. New York: Van Nostrand Reinhold, 1992. ISBN 0-442-01272-01.
- ________. "Standardization of Color Image Data Compression". Part I. "Sequential Coding". *Proceedings Electronic Imaging '89 East* (October 2–5, 1989): 109–112.
- *TIFF*. Revision 6.0, Final. Aldus Corp.: June 3, 1992.

- Welch, Terry A. “A Technique for High Performance Data Compression”. *IEEE Computer*, vol. 17, no. 6 (June 1984).

These publications describe color and grayscale images:

- Commission Internationale de l’Eclairage. *Colorimetry*. CIE Publication no. 15-2.
- Hunt, R. *The Representation of Colour in Photography, Printing and Television* 5th ed. Foundation Press, 1995.
- Lucky, R. W., J. Salz, and E. J. Weldon Jr. *Principles of Data Communication* (New York: McGraw-Hill, 1968).

Index

InfoPrint Solutions Company™

Printed in USA

S550-1142-00