PCL XL Feature Reference Protocol Class 2.1 Supplement

Revision: 1.0
Revision Date: August 9, 2000
Author:
Word for Windows File: xl_ref21.doc

Document Revision History

Rev	Revision Description	Date	Author
1.0	Release	09Aug2000	

***** NOTICE *****

HEWLETT-PACKARD COMPANY MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING THE SOFTWARE OR TECHNICAL INFORMATION. HEWLETT-PACKARD COMPANY DOES NOT WARRANT, GUARANTEE OR MAKE ANY REPRESENTATIONS REGARDING THE USE OR THE RESULTS OF THE USE OF THE SOFTWARE OR TECHNICAL INFORMATION IN TERMS OF ITS CORRECTNESS, ACCURACY, RELIABILITY, CURRENTNESS, OR OTHERWISE. YOU ASSUME THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE SOFTWARE OR TECHNICAL INFORMATION. The exclusion of implied warranties is not permitted by some jurisdictions. The above exclusion may not apply to you.

IN NO EVENT WILL HEWLETT-PACKARD COMPANY BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, INCIDENTAL OR INDIRECT DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION AND THE LIKE) ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE OR TECHNICAL INFORMATION EVEN IF HEWLETT-PACKARD HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Because some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitations may not apply to you. Hewlett-Packard liability to you for actual damages from any cause whatsoever, and regardless of the form of the action (whether in contract, tort including negligence, product liability or otherwise), will be limited to US $50.

Table Of Contents

1.0 Introduction

This document is designed to provide a supplement to the ***PCL-XL Protocol 2.0 Reference Manual***. Information in this supplement is on operators that have changes for protocol class 2.1 as well as information about new operators. Information that is different from the 2.0 reference manual is printed in red for clarity.

2.0 PCL XL Operators, Attributes and Streams

OPERATOR: BEGINPAGE

Protocol Class:

1.0

Purpose

Begin the definition of a new page.

Precondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

Attribute List Specification

multiAttributeList ::= Orientation $_{opt}$ &
{ MediaSize $_{opt}$ | {CustomMediaSize & CustomMediaSizeUnits} $_{opt}$ } &
MediaType$_{opt}$ &
MediaSource$_{opt}$ &
MediaDestination$_{opt}$ &
{ SimplexPageMode $_{opt}$ | {DuplexPageMode $_{opt}$ & {DuplexPageSide $_{opt}$ } }

Attribute ID	Description and {value}
Orientation	The orientation of the logical page. If Orientation has not been set by a BeginPage operator, the device's default setting will be used for the first page, or the previous page's orientation will be used for subsequent pages. {ePortraitOrientation | eLandscapeOrientation | eReversePortrait | eReverseLandscape}
MediaSize	An enumeration for size of the target surface. { see attribute enumeration table below } or a case sensitive string matching an entry in the current printer's list of supported paper sizes. If MediaSize has not been set by a BeginPage operator, the device's default setting will be used for the first page, or the previous page's size will be used for subsequent pages. { see attribute enumeration table in Appendix __ }
CustomMediaSize	The dimensions of a custom media size. { xy_value }
CustomMediaSizeUnits	An enumeration for the unit of measure for the custom media size dimensions. { eInch | eMillimeter | eTenthsOfAMillimeter }

MediaType	A string "name". The attribute data type of ubyte_array is used to send the string name of the media type desired. {ubyte_array}
MediaSource	The source from which the media will be pulled for a physical-media device. If not included, the device will use the most recent MediaSource setting by a BeginPage operator. If MediaSource has not been set by a BeginPage operator, the device's default setting will be used. { see attribute enumeration table in Appendix __ }
MediaDestination	The destination for each page after it is printed. If not included, the device will use the most recent MediaSestination setting by a BeginPage operator. If MediaDestination has not been set by a BeginPage operator, the device's default setting will be used. { see attribute enumeration table in Appendix __ }
SimplexPageMode	An enumeration requesting that pages be rendered one page per physical media page and rendered on the front side of each physical media page. If not included, the device will use the most recent simplex/duplex setting by a BeginPage operator. If simplex/duplex has not been set by a BeginPage operator, the device will use the device default setting. {eSimplexFrontSide}
DuplexPageMode	An enumeration requesting that pages be rendered on both sides of a physical media page and oriented for either horizontal or vertical binding of the physical pages. If not included, the device will use the most recent simplex/duplex setting by a BeginPage operator. If simplex/duplex has not been set by a BeginPage operator, the device will use the device default setting. {eDuplexHorizontalBinding | eDuplexVerticalBinding}
DuplexPageSide	An enumeration specifying the side of the media on which the current page should be rendered. If not included, the device will use the most recent simplex/duplex setting by a BeginPage operator. If simplex/duplex has not been set by a BeginPage operator, the device will use the device default setting. {eFrontMediaSide | eBackMediaSide}

Postcondition

The device has been put into a mode in which painting operators may be accepted.

The graphics state attributes have been initialized to default values.

Example

Beginning a simple page. Use the default paper size and orientation

```
BeginPage
```

Beginning a simple page using Letter size paper.

```
ubyte eLetterPaper MediaSize
BeginPage
```

Beginning a simple page on letter size paper in lasndscape mode.

```
ubyte eLandscapeOrientation Orientation
ubyte eLetterPaper MediaSize
BeginPage
```

Beginning a page printing on Executive sized paper with duplexing enabled.

```
ubyte ePortraitOrientation Orientation
ubyte eExecPaper MediaSize
ubyte eDuplexVerticalBinding DuplexPageMode
BeginPage
```

Comments

See **Appendix F** and **Appendix G** for valid attribute data types and ranges.

See **Appendix K** for related error codes.

MediaSize Notes:

The media size for all pages in a job that do not have a MediaSize attribute attached to the BeginPage operator is always the same size as the previous page. If the page is the first page of the job and no MediaSize is specified or MediaSize is set to eDefaultMediaSize: XL uses the currently set default paper size (set via the front panel or PJL) for this and all following pages of the job until another MediaSize is specified.

MediaSize can be selected by enumeration or by sending a string name for the size desired. Every device stores a list of supported media sizes that includes the code numbers, the size information, and a name string. By scanning the device's list for a name matching the one sent to PCL XL by the user, the correct enumeration for that size can be obtained by XL and from that PCL XL gains the information on size and format. This way XL can support paper sizes it currently knows nothing about but that future devices support.

There are five cases in which PCL XL selects the paper size for the user. These cases are:

- When no MediaSize is specified or eDefaultMediaSize is specified.
- The MediaSize specified is not a valid enumeration value.
- The MediaSize specified is a valid enumeration but not supported by the device.
- The MediaSize string name sent does not match any name in the device's list of supported sizes.
- The device does not support a user defined custom paper size.

In the first four cases PCL XL will use the default paper size specified by front-panel or PJL settings.

In the last case, when the user sends down the dimension for a custom sized page, PCL XL will query the device to determine if that size is supported. If the device responds with an OK, then the page is processed normally. If the device responds that the dimensions are not supported, PCL XL will search the devices media size list for a standard size that most closely matches the desired size, select that media size, print, and issue a warning.

Example

Selecting paper size by string

```
ubyte_array (Letter) MediaSize
```

Selecting a paper size by enumeration

```
ubyte_array eLetterPaper MediaSize
```

MediaType Notes:

PCL XL normally passes the media type name to the device without checking validity of the type. There are no PCL XL warnings or errors associated with media types. There are two cases where PCL XL will respond to the media type the user defines.

- If the media type is "Transparency", PCL XL will automatically disable duplexing until a non-transparency media type is specified.
- If the media type is "PrePunch" and duplexing is enabled, and the printer is a landscape feed device, PCL XL will rotate the image on the page 180° before printing to ensure that the punched holes end up on the left edge of the page.

For HP LaserJet Printer devices, the MediaType attribute is persistent through the end of the session unless overridden by another MediaType setting on a successive page.

Example

Selecting a media type for letterhead paper.

```
ubyte_array (Letterhead) MediaType
```

Selecting media type as plain, use an empty (NULL) string

```
ubyte_array () MediaType
```

MediaSource Notes:

There are three cases in which PCL XL selects the paper source for the user. These cases are:

- The user does not specify a media source in BeginPage.
- The media source specified is not a valid enumeration value.
- The media source is a valid enumeration but not supported by the device.

In all cases, PCL XL will choose the last valid media source selected by a previous BeginPage, or the default media source if the current page is the first page in the session. The default media source is specified by front-panel settings or via PJL.

For HP LaserJet Printer devices, the MediaSource attribute is persistent through the end of the session unless overridden by another MediaSource setting on a successive page.

Example

Allowing the printer to select a tray based on media size, type, etc.

```
ubyte eAutoSelect MediaSource
```

Selecting the manual feed tray for input.

```
ubyte eManualFeed MediaSource
```

Selecting an external input device's first tray (all external input trays are numbered starting with 8).

```
ubyte 8 MediaSource
```

MediaDestination Notes:

There are three cases in which PCL XL selects the paper destination for the user. These cases are:

- The user does not specify a media destination BeginPage,
- The media destination specified is not a valid enumeration value, and
- The media destination is a valid enumeration but not supported by the device.

In all cases, PCL XL will choose the last valid media destination selected by a previous BeginPage or the default media destination if the current page is the first page in the session. The default media destination is specified by front-panel settings or via PJL.

For HP LaserJet Printer devices, the MediaDestination attribute is persistent through the end of the session unless overridden by another MediaDestination setting on a successive page.

Example

Selecting the face up bin on printers so equipped.

```
ubyte eFaceUpBin MediaDestination
```

Selecting an external device's first output bin (all external output bins are numbered starting with 5).

```
ubyte 5 MediaDestination
```

Duplexing Notes:

If the output device is in duplex mode and a change in MediaSource, MediaDestination, MediaSize, MediaType or DuplexBinding occurs in BeginPage for the front side of a page, the device will send one blank page to finish the duplexed page. The new front page has the new BeginPage attributes.

If the output device is in duplex mode and a change for MediaSource, MediaSize, MediaType, or DuplexBinding occurs in BeginPage for the backside of a page, the device will output two blank pages. These pages: 1) finish the back side for the current page and 2) provide a front side for the new back page having the new BeginPage attributes.

DuplexPageMode Notes:

PJL's long edge and short edge binding refer to the physical paper, while XL's eDuplexHorizontalBinding eDuplexVerticalBinding refer to the page's image orientation (eLandscapeOrientation, ePortraitOrientation, eReverseLandscape, or eReversePortrait). Example, When eDuplexVerticalBinding is specified, the binding is the paper's long edge on a ePortraitOrientation page, and short the edge on a eLandscapeOrientation page.

In addition, XL ignores PageAngle when it calculates the binding because PageAngle is 0 degrees (default) when the BeginPage operator is executed.

OPERATOR: BEGINIMAGE

Purpose

Setup the device for painting a new bitmap image at the current cursor location.

Precondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

The data source must be open from which image blocks are to be read.

The current color space must match the image data.

If the image pixel colors are indexed to a palette, the palette must exist in the color space and the palette length must be compatible with the depth of the image source pixels.

The current cursor must be set to a valid x, y point.

Neither the source width nor source height is zero.

Neither the x nor y destination size value is zero.

Attribute List Specification

multiAttributeList ::= { ColorDepth & ColorMapping &
DestinationSize & SourceWidth & SourceHeight }

Attribute ID	Description and {value}
ColorMapping	An enumeration specifying whether the component color mapping is direct or indexed through a palette. { eDirectPixel | eIndexedPixel }
ColorDepth	The number of bits per image component. { e1Bit | e4Bit | e8Bit }
SourceWidth	The width of the image source in source pixels. { +integer }
SourceHeight	The height of the image source in source pixels. { +integer }
DestinationSize	The size of the destination box on the page for the image in current user units. { xyValue }

Postcondition

The device has been set up to accept ReadImage operators.

Example

```
ubyte eDirectPixel ColorMapping
```

```
ubyte e8Bit ColorDepth
uint16 640 SourceWidth
uint16 480 SourceHeight
uint16_xy 3200 2400 DestinationSize
BeginImage
```

Comments

See **Appendix F** and **Appendix G** for valid attribute data types and ranges.

See **Appendix K** for related error codes.

OPERATOR: READIMAGE

Purpose

Read a block of a bitmap image data from the current data source.

Precondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

The StartLine attribute is in sequence relative to the end line of the previous ReadImage.

The data source from which image blocks are read must be open.

The data source must have at least one block remaining that matches the BeginImage attribute list definition and the attribute list definition for this ReadImage.

The first byte of the image block is the left-most byte of the block (No byte swapping is performed on the binary data describing the image block).

The length of each line of image data within the image block must be a multiple of four bytes unless the PadBytesMultiple parameter is specified. If the PadBytesMultiple attribute is supplied the number of bytes in each line must be a multiple of the value of that attribute.

Attribute List Specification

multiAttributeList ::= { CompressMode & StartLine & BlockHeight
& {PadBytesMultiple}$_{opt}$ & {BlockByteLength}$_{opt}$}

Attribute ID	Description and {value}
StartLine	The line within the source image at which this block is located. The line position is in source pixels. The line must be in sequence relative to the end line of the previous ReadImage. The first start line must be zero. { +integer }
BlockHeight	The height of the image data block in source pixels. { +integer }
CompressMode	The compression mode for the block being read. { eNoCompression | eRLECompression | eJPEGCompression | eDeltaRowCompression}
PadBytesMultiple	The number of pad bytes for each line in the image block (default = 4). This number must be in the range 1 to 4. {+ubyte}

BlockByteLength	This attribute specifies the number of bytes used to described the image block. If the block is compressed, the number is the number of bytes in compressed form. If this attribute is not included the source image data is embedded within the XL stream and the embedded data tag describes the number of bytes in the block. {+integer}

Postcondition

A block of bitmap image data has been read by the device from the current data source.

Example

```
uint16 0 StartLine
uint16 480 BlockHeight
ubyte eNoCompression CompressMode
// Operator Position: 10
ReadImage

dataLength 921600
hex_raw* [
34 55 6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d 34
55 6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d 34 55
.
.
.
6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d
34 55 6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d 34
[
```

Comments

The bitmap image is affected by the current brush and ROP settings. The brush is considered the “Pattern” and the bitmap is considered the “source”. If a ROP is set that uses both “Pattern” and “source” the appropriate ROP is executed, affecting the final image that is placed on the page. For example, if the brush source is RGB blue and the default ROP is set then the bitmap will be OR’d with the brush source.

While it is permitted to mix eRLECompression and eNoCompression blocks of ReadImage data, XL does not allow mixing JPEG or DeltaRow image blocks with any other compression method.

See **Appendix F** and **Appendix G** for valid attribute data types and ranges.

See **Appendix K** for related error codes.

OPERATOR: ENDIMAGE

Purpose

End the definition of a bitmap image and cause it to be painted.

Precondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

Attribute List Specification

nullAttributeList

Postcondition

The newly defined image has been scheduled for painting.

Example

```
EndImage
```

Comments

See **Appendix K** for related error codes.

OPERATOR: BEGINRASTPATTERN

Purpose

Start the definition of a new raster pattern (bitmap pattern).

Precondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

The data source from which raster pattern blocks are to be read must be open.

Neither the source width nor source height is zero.

Neither the x nor y destination size value is zero.

Attribute List Specification

multiAttributeList ::= { ColorDepth & ColorMapping &
DestinationSize & SourceWidth & SourceHeight &
{ PatternDefineID & PatternPersistence } }

Attribute ID	Description and {value}
ColorMapping	An enumeration specifying whether the component color mapping is direct or indexed through a palette. { eDirectPixel | eIndexedPixel }
ColorDepth	The number of bits per image component. { e1Bit | e4Bit | e8Bit }
SourceWidth	The width of the raster pattern source in source pixels. { +integer }
SourceHeight	The height of the raster pattern source in source pixels. { +integer }
DestinationSize	The size of the destination box on the page for the raster pattern specified in user units. { xyValue }
PatternDefineID	The numeric identifier by which the pattern will be known and selected in a SetBrushSource and/or SetPenSource operation. The same id number may be reused for temporary-, page-, and session-persistent patterns without destroying the previously defined patterns. However, the precedence on pattern selection in SetPen/BrushSource is temporary first (if exists), page second (if exists), and session third (if exists). {integer}

PatternPersistence	An enumerated value specifying the persistence of the pattern. If the enumeration is eTempPattern, the pattern will be created with temporary persistence. The pattern is accessible by a SetPenSource or SetBrushSource using this identifier in the current or higher (PushGS) graphics state levels until replaced by a temporary pattern using the same identifier or until the current graphics state level is popped (PopGS). If the enumeration is ePagePattern the pattern will be created with page persistence. The pattern is accessible by a SetPenSource or SetBrushSource using the identifier for the remainder of the page or until replaced by a page pattern using the same identifier. If the enumeration is eSessionPattern, the pattern will be created with session persistence. The pattern is accessible by a SetPenSource or SetBrushSource using the identifier for the remainder of the session or until replaced by a session pattern using the same identifier. { eTempPattern | ePagePattern | eSessionPattern }

Postcondition

A raster pattern descriptor has been defined and associated to a pattern identifier. No pattern data yet exists.

Any pattern defined with the same pattern ID at the same persistence level will have been deleted.

Example

```
ubyte eDirectPixel ColorMapping
ubyte e8Bit ColorDepth
uint16 8 SourceWidth
uint16 8 SourceHeight
uint16_xy 8 8 DestinationSize
ubyte ePagePattern PatternPersistance
BeginRastPattern
```

Comments

Note: A problem has been identified with the PCL XL 1.x pattern subsystem that requires a rule be followed when designing for backward compatibility with PCL XL 2.0. For PCL XL 1.x, the color space in effect when a pattern is set to a pen or a brush must be the same color space and palette that was in effect at the time the pattern was originally downloaded.

See **Appendix F** and **Appendix G** for valid attribute data types and ranges.

See **Appendix K** for related error codes.

OPERATOR: READRASTPATTERN

Purpose

Read a block of a raster pattern data from the current data source.

Precondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

The data source from which raster pattern blocks are read must be open.

The StartLine attribute is in sequence relative to the end line of the previous ReadRastPattern.

The data source must have at least one block remaining that matches the attribute list definition for BeginRastPattern and the attribute list definition for this ReadRastPattern.

The first byte of the pattern block is the left-most byte of the block (no byte swapping is performed on the binary data describing the raster pattern block).

The length of each line of image data within the pattern block must be a multiple of four bytes. If a line is not a multiple of four bytes, it must be padded with the appropriate number of bytes with a value of zero.

The length of each line of pattern data within the pattern block must be a multiple of four bytes unless the PadBytesMultiple parameter is specified. If the PadBytesMultiple attribute is supplied the number of bytes in each line must be a multiple of the value of that attribute.

Attribute List Specification

multiAttributeList ::= { CompressMode & StartLine & BlockHeight
& {PadBytesMultiple}$_{opt}$ & {BlockByteLength}$_{opt}$}

Attribute ID	Description and {value}
StartLine	The line within the source raster block to start reading data in source pixels. The line must be in sequence relative to the end line of the previous ReadImage. The first start line must be zero. { +integer }
BlockHeight	The height of the raster data block in source pixels. { +integer }
CompressMode	The compression mode for the block being read. { eNoCompression | eRLECompression | eJPEGCompression | eDeltaRowCompression}
PadBytesMultiple	The number of pad bytes for each line in the pattern image block (default = 4). This number must be greater than zero. {+ubyte}

BlockByteLength	This attribute specifies the number of bytes used to described the pattern image block. If the block is compressed, the number is the number of bytes in compressed form. If this attribute is not included the source image data is embedded within the XL stream and the embedded data tag describes the number of bytes in the block. {+integer}

Postcondition

A block of raster pattern data has been read by the device from the current data source.

Example

```
uint16 0 StartLine
uint16 8 BlockHeight
ubyte eNoCompression CompressMode
// Operator Position: 10
ReadRastPattern

dataLength 64
hex_raw* [
34 55 6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d 34
55 6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d 34 55
6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d
34 55 6d 34 55 6d 34 55 6d 34 55 6d 34 55 6d 34
[
```

Comments

See **Appendix F** and **Appendix G** for valid attribute data types and ranges.

See **Appendix K** for related error codes.

OPERATOR: ENDRASTPATTERN

Purpose

End the definition of a raster pattern. The pattern defined is ready for use during painting operations by assignment to the Pen or Brush.

Precondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

Attribute List Specification

nullAttributeList

Postcondition

A new raster pattern has been defined for a pen and/or a brush.

Example

```
EndRastPattern
```

Comments

See **Appendix K** for related error codes.

OPERATOR: SETDEFAULTGS

Protocol Class:

2.1

Purpose

Sets the current graphics state to its default (BeginPage) condition.

PreCondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

Attribute List Specification

NullAttributeList

Precondition

The current graphics state object has been set to its BeginPage default state.

Example

```
SetDefaultGS
```

Comments

See **Appendix K** for related error codes.

In PCL-XL 2.0, there was no way to restore the graphics state to default after any modifications were done. This could be a problem for, among other things, Streams (macros). Once a modified GS was pushed on the stack, there was no way to go back to known settings. With this new operator the user can do a PushGS, SetDefaultGS, execute a stream or whatever needs a base GS to operate, then a PopGS will return the modified GS.

See **Appendix K** for related error codes.

Comments

OPERATOR: SETCOLORTREATMENT

Protocol Class

2.1

Purpose

Selects the color treatment to be applied to RGB colors.

Precondition

A color space of eRGB has been established by default, or by executing a SetDefaultGS or SetColorSpace operator.

Attribute List Specification

multiAttributeList ::= ColorTreatment

Attribute ID	Description and {value}
ColorTreatment	The color treatment enumeration to be applied to RGB color processing. { eNoTreatment | eVivid | eScreenMatch }

Postcondition

The color treatment for the current color space has been set.

Example

```
Ubyte eScreenMatch ColorTreatment
SetColorTreatment
```

Comments

See **Appendix F** and **Appendix G** for valid attribute data types and ranges.

See **Appendix K** for related error codes.

OPERATOR: EXECSTREAM

Purpose

Immediately execute a previously defined stream that contains a sequence of PCL XL operators and data.

Precondition

Immediate execution of this operator occurs in a legal PCL XL operator sequence.

The stream contains legal PCL XL attribute lists and operators.

There is not a stream of the same name already running.

Attribute List Specification

multiAttributeList ::= StreamName

Attribute ID	Description and {value}
StreamName	The name of the user-defined stream to be executed. { integerArray }

Postcondition

The stream associated with the stream name has begun execution.

Example

```
ubyte_array (SampleStream) StreamName
ExecStream
```

Comments

If no stream named by StreamName exists, PCL XL will raise an error.

There are three rules that control when one stream can invoke another.

Rule 1 - The stream nesting limit is 32.

Rule 2 – A stream may not invoke another copy of itself.

Rule 3 – A stream may not invoke another copy of any other stream that is already running.

Violation of any of these rules will raise an error.

See **Appendix F** and **Appendix G** for valid attribute data types and ranges.

See **Appendix K** for related error codes.

5.0 Graphics State Settings

Graphics state attributes are set by the user to obtain a specific result during painting. For example, the graphics state contains the current paint source (color) associated with the brush. When an object is painted, the device retrieves the brush's current paint source from the graphics state. If the brush is associated with a valid paint source PCL XL fills the object with the corresponding color or pattern specified.

DEFAULT GRAPHICS STATE

All graphics state attributes are set by commands preceded by "**Set**..." with the exception of the current path which is set by path operators (i.e. Begin/EndPath, Arc, Line, etc.). Graphics state attributes and defaults at each **BeginPage**, or when the **SetDefaultGS** operator is executed are listed in the table below:

GS Attribute	Description	Default
BrushSource	Paint source currently associated with the brush	RGB black
ColorTreatment	The color treatment to be applied to the current RGB colors.	eScreenMatch
CharAngle	The angle at which to draw characters (additive to page CTM)	0
CharScale	The scaling factor for characters (additive to page CTM)	x=1, y=1
CharShear	The shearing factor for characters (additive to page CTM)	x=0, y=0
ClipMode	The mode determining even-odd or non-zero winding construction of the clip path	eNonZeroWinding
ColorSpace	Current color space	RGB
CurrentClipPath	The region defining the current clip path	imagable area of the page
CurrentFont	The font that will be used for painting characters	no font defined
CurrentPath	The region defining the current path	no path defined
DefaultCTM	The default page coordinate transformation matrix	Device dependent
DitherAnchor	The x,y point in which the dither is anchored	0,0
DitherMatrixID	A user definable identifier used to select a specific dither matrix	none
HalftoneMethod	The method used for halftone operations	DitherMatrix eDeviceBest
FillMode	The mode in which closed paths should be filled	eNonZeroWinding
LineCap	The shape to draw on the end of lines (open subpaths)	eButtCap
LineDash	The dash style to use when stroking lines with a pen	solid line
LineJoin	The shape to draw where two lines meet at an angle	eMiterJoin
MiterLimit	The length limit on miter join shapes	10.0
PageCTM	Current page coordinate transformation matrix	session defaults at the current page orientation
PaintTxMode	The current transparency mode for patterns	eOpaque
PalletteID	A user definable identifier used to select a specific palettes	none
PatternAnchor	The x,y point in which the brush and pen patterning is started	0,0
PenSource	Paint source currently associated with the pen	RGB black

PenWidth	The current width (in user units) for pen stroking operations	1 user unit
ROP	The current raster operation in effect	ROP3=252
SourceTxMode	The current transparency mode for source objects	eOpaque

Appendices

Appendix B. Binary Stream Tag Values

The following table outlines the specific values assigned for attribute list tags, operator tags, and data type tags.

Tag Value	Tag Name	Tag Type	Description
0x00		White Space	Null
0x01 - 0x08			Not Used
0x09 - 0x0d		White Space	HT, LF, VT, FF, CR
0x0e - 0x1f			Not Used
0x20		White Space	Space
0x21 - 0x26			Not Used
0x27			Reserved for beginning of ASCII binding.
0x28			Binary binding - high byte first.
0x29			Binary binding - low byte first.
0x2a - 0x40			Not Used
0x41	BeginSession	Operator	
0x42	EndSession	Operator	
0x43	BeginPage	Operator	
0x44	EndPage	Operator	
0x45		Operator	Reserved for future use.
0x46		Operator	Reserved for future use.
0x47	Comment	Operator	
0x48	OpenDataSource	Operator	
0x49	CloseDataSource	Operator	
0x4a		Operator	Reserved for future use.
0x4b		Operator	Reserved for future use.
0x4c		Operator	Reserved for future use.
0x4d		Operator	Reserved for future use.
0x4e		Operator	Reserved for future use.
0x4f	BeginFontHeader	Operator	
0x50	ReadFontHeader	Operator	
0x51	EndFontHeader	Operator	
0x52	BeginChar	Operator	
0x53	ReadChar	Operator	
0x54	EndChar	Operator	
0x55	RemoveFont	Operator	
0x56	SetCharAttributes	Operator	

0x57	SetDefaultGS	Operator	
0x58	SetColorTreatment	Operator	
0x59		Operator	
0x5a		Operator	
0x5b	BeginStream	Operator	
0x5c	ReadStream	Operator	
0x5d	EndStream	Operator	
0x5e	ExecStream	Operator	
0x5f	RemoveStream	Operator	
0x60	PopGS	Operator	
0x61	PushGS	Operator	
0x62	SetClipReplace	Operator	
0x63	SetBrushSource	Operator	
0x64	SetCharAngle	Operator	
0x65	SetCharScale	Operator	
0x66	SetCharShear	Operator	
0x67	SetClipIntersect	Operator	
0x68	SetClipRectangle	Operator	
0x69	SetClipToPage	Operator	
0x6a	SetColorSpace	Operator	
0x6b	SetCursor	Operator	
0x6c	SetCursorRel	Operator	
0x6d	SetHalftoneMethod	Operator	
0x6e	SetFillMode	Operator	
0x6f	SetFont	Operator	
0x70	SetLineDash	Operator	
0x71	SetLineCap	Operator	
0x72	SetLineJoin	Operator	
0x73	SetMiterLimit	Operator	
0x74	SetPageDefaultCTM	Operator	
0x75	SetPageOrigin	Operator	
0x76	SetPageRotation	Operator	
0x77	SetPageScale	Operator	
0x78	SetPatternTxMode	Operator	
0x79	SetPenSource	Operator	
0x7a	SetPenWidth	Operator	
0x7b	SetROP	Operator	
0x7c	SetSourceTxMode	Operator	
0x7d	SetCharBoldValue	Operator	
0x7f	SetClipMode	Operator	
0x80	SetPathToClip	Operator	
0x81	SetCharSubMode	Operator	
0x82	BeginUserDefinedLineCaps	Operator	
0x83	EndUserDefinedLineCaps	Operator	
0x84	CloseSubPath	Operator	
0x85	NewPath	Operator	
0x86	PaintPath	Operator	
0x87		Reserved	
0x88		Reserved	
0x89		Reserved	
0x8a		Reserved	
0x8b		Reserved	

0x8c		Operator	
0x8d		Operator	
0x8e		Operator	
0x8f		Operator	
0x90		Operator	Reserved for future use.
0x91	ArcPath	Operator	
0x92		Operator	Reserved for future use.
0x93	BezierPath	Operator	
0x94		Operator	Reserved for future use.
0x95	BezierRelPath	Operator	
0x96	Chord	Operator	
0x97	ChordPath	Operator	
0x98	Ellipse	Operator	
0x99	EllipsePath	Operator	
0x9a		Operator	Reserved for future use.
0x9b	LinePath	Operator	
0x9c		Operator	Reserved for future use.
0x9d	LineRelPath	Operator	
0x9e	Pie	Operator	
0x9f	PiePath	Operator	
0xa0	Rectangle	Operator	
0xa1	RectanglePath	Operator	
0xa2	RoundRectangle	Operator	
0xa3	RoundRectanglePath	Operator	
0xa4		Operator	Reserved for future use.
0xa5		Operator	Reserved for future use.
0xa6		Operator	Reserved for future use.
0xa7		Operator	Reserved for future use.
0xa8	Text	Operator	
0xa9	TextPath	Operator	
0xaa		Operator	Reserved for future use.
0xab		Operator	Reserved for future use.
0xac		Operator	Reserved for future use.
0xad		Operator	Reserved for future use.
0xae		Operator	Reserved for future use.
0xaf		Operator	Reserved for future use.
0xb0	BeginImage	Operator	
0xb1	ReadImage	Operator	
0xb2	EndImage	Operator	
0xb3	BeginRastPattern	Operator	
0xb4	ReadRastPattern	Operator	
0xb5	EndRastPattern	Operator	
0xb6	BeginScan	Operator	
0xb7		Operator	Reserved for future use.
0xb8	EndScan	Operator	
0xb9	ScanLineRel	Operator	
0xba - 0xbf			Reserved for future use.
0xc0	ubyte	Data Type	Unsigned 8-bit value
0xc1	uint16	Data Type	Unsigned 16-bit value
0xc2	uint32	Data Type	Unsigned 32-bit value

0xc3	sint16	Data Type	Signed 16-bit value
0xc4	sint32	Data Type	Signed 32-bit value
0xc5	real32	Data Type	Real number value
0xc6		Data Type	Reserved for future use.
0xc7		Data Type	Reserved for future use.
0xc8	ubyte_array	Data Type	Array of Unsigned 8-bit values
0xc9	uint16_array	Data Type	Array of Unsigned 16-bit values
0xca	uint32_array	Data Type	Array of Unsigned 32-bit values
0xcb	sint16_array	Data Type	Array of Signed 16-bit values
0xcc	sint32_array	Data Type	Array of Signed 32-bit values
0xcd	real32_array	Data Type	Array of Real number values
0xce		Data Type	Reserved for future use.
0xcf		Data Type	Reserved for future use.
0xd0	ubyte_xy	Data Type	Two Unsigned 8-bit values
0xd1	uint16_xy	Data Type	Two Unsigned 16-bit values
0xd2	uint32_xy	Data Type	Two Unsigned 32-bit values
0xd3	sint16_xy	Data Type	Two Signed 16-bit values
0xd4	sint32_xy	Data Type	Two Signed 32-bit values
0xd5	real32_xy	Data Type	Two Real number values
0xd6-0xdf		Data Type	Reserved for future use.
0xe0 - 0xf7			Reserved for future use.
0xf8	attr_ubyte	Attribute	Unsigned, 8-bit Attribute
0xf9	attr_uint16	Attribute	Unsigned, 16-bit Attribute
0xfa	dataLength	Embed Data	Embedded Data Follows
0xfb	dataLengthByte	Embed Data	Embedded Data Follows (0-255 bytes)
0xfc - 0xff			Reserved for future use.

Appendix E. Attribute ID Number to Attribute Name Table

The following table outlines the attribute ID numbers for PCL XL attributes in binary streams.

Attribute ID	Attribute Name
1.	
2.	PaletteDepth
3.	ColorSpace
4.	NullBrush
5.	NullPen
6.	PaletteData
7.	
8.	PatternSelectID
9.	GrayLevel
10.	
11.	RGBColor
12.	PatternOrigin
13.	NewDestinationSize
14.	PrimaryArray
15.	PrimaryDepth
16.	
17.	
18.	
19.	
20.	
21.	
22.	
23.	
24.	
25.	
26.	
27.	
28.	
29.	
30.	
31.	
32.	
33.	DeviceMatrix
34.	DitherMatrixDataType
35.	DitherOrigin
36.	MediaDestination
37.	MediaSize
38.	MediaSource
39.	MediaType
40.	Orientation
41.	PageAngle
42.	PageOrigin
43.	PageScale
44.	ROP3
45.	TxMode
46.	

47.	CustomMediaSize
48.	CustomMediaSizeUnits
49.	PageCopies
50.	DitherMatrixSize
51.	DitherMatrixDepth
52.	SimplexPageMode
53.	DuplexPageMode
54.	DuplexPageSide
55.	
56.	
57.	
58.	
59.	
60.	
61.	
62.	
63.	
64.	
65.	ArcDirection
66.	BoundingBox
67.	DashOffset
68.	EllipseDimension
69.	EndPoint
70.	FillMode
71.	LineCapStyle
72.	LineJoinStyle
73.	MiterLength
74.	LineDashStyle
75.	PenWidth
76.	Point
77.	NumberOfPoints
78.	SolidLine
79.	StartPoint
80.	PointType
81.	ControlPoint1
82.	ControlPoint2
83.	ClipRegion
84.	ClipMode
85.	
86.	
87.	
88.	
89.	
90.	
91.	
92.	
93.	
94.	
95.	
96.	
97.	
98.	ColorDepth

99.	BlockHeight
100.	ColorMapping
101.	CompressMode
102.	DestinationBox
103.	DestinationSize
104.	PatternPersistence
105.	PatternDefineID
106.	
107.	SourceHeight
108.	SourceWidth
109.	StartLine
110.	PadBytesMultiple
111.	BlockByteLength
112.	
113.	
114.	
115.	NumberOfScanLines
116.	
117.	
118.	
119.	
120.	ColorTreatment
121.	
122.	
123.	
124.	
125.	
126.	
127.	
128.	
129.	CommentData
130.	DataOrg
131.	
132.	
133.	
134.	Measure
135.	
136.	SourceType
137.	UnitsPerMeasure
138.	
139.	StreamName
140.	StreamDataLength
141.	
142.	
143.	ErrorReport
144.	
145.	Reserved
146.	Reserved
147.	Reserved
148.	Reserved
149.	Reserved
150.	Reserved

151.	Reserved
152.	Reserved
153.	Reserved
154.	Reserved
155.	Reserved
156.	Reserved
157.	Reserved
158.	Reserved
159.	Reserved
160.	
161.	CharAngle
162.	CharCode
163.	CharDataSize
164.	CharScale
165.	CharShear
166.	CharSize
167.	FontHeaderLength
168.	FontName
169.	FontFormat
170.	SymbolSet
171.	TextData
172.	CharSubModeArray
173.	
174.	
175.	XSpacingData
176.	YSpacingData
177.	CharBoldValue
178.	
179.	

Appendix F. Attribute Name to Data Types Table

The following table outlines the attribute ID numbers for PCL XL attributes in binary streams.

Attribute Name	Valid Value(s)	Data Types	Attribute #	Class.rev #
ArcDirection	ArcDirection Enumeration	ubyte	65	1.1
BlockByteLength	0 - (2^{32} - 1)	uint32	111	2.0
BlockHeight	0 - 65535	uint16	99	1.1
BoldValue	0.0 - 1.0	real32	177	1.1
BoundingBox	Range of data types	ubyte_box uint16_box sint16_box	66	1.1
CharAngle	-360 <= CharAngle <= 360	uint16 sint16 real32	161	1.1
CharCode	Any value within Range of data types	ubyte uint16	162	1.1
CharDataSize	0 - 65535	uint16	163	1.1
CharScale	Two numeric values less than or equal to 32767.0 and greater than or equal to -32768.0, excluding a value of zero.	ubyte_xy uint16_xy real32_xy	164	1.1
CharShear	Two numeric values within Range of data types and less than 32767.0 and greater than -32768.0	ubyte_xy uint16_xy sint16_xy real32_xy	165	1.1
CharSize	Any numeric value within range of data types but less than 32767.0 and greater than zero.	ubyte uint16 real32	166	1.1
CharSubModeArray	CharSubModeArray Enumeration	ubyte_array	172	1.1
ClipMode	ClipMode Enumeration	ubyte	84	1.1
ClipRegion	ClipRegion Enumeration	ubyte	83	1.1
ColorDepth	ColorDepth Enumeration	ubyte	98	1.1
ColorMapping	ColorMapping Enumeration	ubyte	100	1.1
ColorSpace	ColorSpace Enumeration	ubyte	3	1.1
ColorTreatment	Enumeration	ubyte	120	2.1
CommentData	Any data	ubyte_array uint16_array	129	1.1
CompressMode	Compress Mode Enumeration	ubyte	101	1.1
ControlPoint1	Range of data types	ubyte_xy uint16_xy sint16_xy	81	1.1

ControlPoint2	Range of data types	ubyte_xy uint16_xy sint16_xy	82	1.1
CustomMediaSize	dimensions of physical custom media size	uint16_xy real32_xy	47	1.1
CustomMediaSizeUnits	Measure Enumeration	ubyte	48	1.1
DashOffset	Range of data types	ubyte uint16 sint16	67	1.1
DataOrg	DataOrg Enumeration	ubyte	130	1.1
DestinationBox	Four numeric values:	uint16_box	102	1.1
DestinationSize	Two numeric values; x nor y equal to zero	uint16_xy	103	1.1
DeviceMatrix	DitherMatrix Enumeration	ubyte	33	1.1
DitherMatrixDataType	eUByte	ubyte	34	1.1
DitherMatrixDepth	e8Bit	ubyte	51	1.1
DitherMatrixSize	1-256	uint16_xy	50	1.1
DitherOrigin	Range of data types	ubyte_xy uint16_xy sint16_xy	35	1.1
DuplexPageMode	DuplexPageMode enumeration	ubyte	53	1.1
DuplexPageSide	DuplexPageSide enumeration	ubyte	54	1.1
EllipseDimension	Range of data types	ubyte_xy uint16_xy	68	1.1
EndPoint	Range of data types	ubyte_xy uint16_xy sint16_xy	69	1.1
ErrorReport	ErrorReport Enumeration	ubyte	143	1.1
FillMode	FillMode Enumeration	ubyte	70	1.1
FontFormat	0	ubyte	169	1.1
FontHeaderLength	0 – 65535	uint16	167	1.1
FontName	Valid Font Name	ubyte_array	168	1.1
GrayLevel	0 – 1.0 if real or range of integer data type	real32 ubyte	9	1.1
LineCapStyle	LineCap Enumeration	ubyte	71	1.1
LineDashStyle	Range of data types Maximum array size = MAXDASHES (device-dependent)	ubyte_array uint16_array sint16_array	74	1.1
LineJoinStyle	LineJoin Enumeration	ubyte	72	1.1
Measure	Measure Enumeration	ubyte	134	1.1
MediaSize	MediaSize Enumeration	ubyte	37	1.1
MediaSource	MediaSource Enumeration	ubyte	38	1.1
MiterLength	range of data types	ubyte uint16	73	1.1
NewDestinationSize	range of data types; neither x nor y equal to zero	uint16_xy	13	1.1
NullBrush	0	ubyte	4	1.1

NullPen	0	ubyte	5	1.1
NumberOfPoints	Range of data types	ubyte uint16	77	1.1
NumberOfScanLines	Range of data types	uint16	115	1.1
Orientation	Orientation Enumeration	ubyte	40	1.1
PadBytesMultiple	1-255	ubyte	110	2.0
PageAngle	Positive or Negative multiples of 90: -360, -270, -180, -90, 0, 90, 180, 270, 360	uint16 sint16	41	1.1
PageCopies	range of data types; zero causes the page not to be imaged	uint16	49	1.1
PageOrigin	Range of data types	ubyte_xy uint16_xy sint16_xy	42	1.1
PageScale	Two numeric values less than or equal to 32767.0 and greater than or equal to 0.	ubyte_xy uint16_xy real32_xy	43	1.1
PaletteData	Any data in the range of the array elements. If eGray, array lengths of 2, 16, and 256 are allowed. If eRGB, array lengths of 6, 48, 768 are allowed.	ubyte_array	6	1.1
PaletteDepth	ColorDepth Enumeration	ubyte	2	1.1
PatternDefineID	Range of data types	sint16	105	1.1
PatternOrigin	Range of data types	sint16_xy	12	1.1
PatternPersistence	PatternPersistence Enumeration	ubyte	104	1.1
PatternSelectID	Range of data types	sint16	8	1.1
PenWidth	Range of data types: zero or more	ubyte uint16	75	1.1
Point	Two numeric values within Range of data types	ubyte_xy uint16_xy sint16_xy	76	1.1
PointType	DataType Enumeration	ubyte	80	1.1
PrimaryArray	Color data for colorspace	real32_array ubyte_array	14	2.0
PrimaryDepth	Primary Depth Enumeration	ubyte	15	2.0
RGBColor	Three values: 0 - 1.0 if real or range of integer data type	real32_array ubyte_array	11	1.1
ROP3	range of data types	ubyte	44	1.1
SimplexPageMode	SimplexPageMode Enumeration	ubyte	52	1.1
SolidLine	0	ubyte	78	1.1
SourceHeight	1 - 65535	uint16	107	1.1
SourceType	DataSource Enumeration	ubyte	136	1.1
SourceWidth	1 - 65535	uint16	108	1.1
StartLine	0 - 65535	uint16	109	1.1

StartPoint	Range of data types	ubyte_xy uint16_xy sint16_xy	79	1.1
StreamDataLength	Range of data types	uint32	140	1.1
StreamName	ASCII character string	ubyte_array uint16_array	139	1.1
SymbolSet	SymbolSet Enumeration	uint16	170	1.1
TextData	Any character codes in range of data types	ubyte_array uint16_array	171	1.1
TxMode	TxMode enumeration	ubyte	45	1.1
UnitsPerMeasure: xUnits, yUnits	two positive numeric values maximum of 65535.0	uint16_xy real32_xy	114	1.1
WritingMode	WritingMode enumeration	Ubyte	173	2.0
XSpacingData	Range of data types—must be same size as TextData array	ubyte_array uint16_array sint16_array	175	1.1
YSpacingData	Range of data types—must be same size as TextData array	ubyte_array uint16_array sint16_array	176	1.1

Appendix G. Attribute Value Enumerations Table

The following are values used for enumerated data types through out this document. These enumerations are standardized across protocol classes. All values are listed in decimal form.

Attribute Name / Enumeration	Value	Class
ArcDirection		
eClockWise	0	1.1
eCounterClockWise	1	1.1
CharSubModeArray		
eNoSubstitution	0	1.1
eVerticalSubstitution	1	1.1
ClipMode		
see FillMode Enumeration		
ClipRegion		
eInterior	0	1.1
eExterior	1	1.1
ColorDepth		
e1Bit	0	1.1
e4Bit	1	1.1
e8Bit	2	1.1
ColorMapping		
eDirectPixel	0	1.1
eIndexedPixel	1	1.1
ColorSpace		
eGray	1	1.1
eRGB	2	1.1
ColorTreatment		
eNoTreatment	0	2.1
eScreenMatch	1	2.1
eVivid	2	2.1
CompressMode		
eNoCompression	0	1.1
eRLECompression	1	1.1
eJPEGCompression	2	2.0
eDeltaRowCompression	3	2.1
DataOrg		
eBinaryHighByteFirst	0	1.1
eBinaryLowByteFirst	1	1.1
DataSource		
eDefault	0	1.1

DataType		
eUByte	0	1.1
eSByte	1	1.1
eUint16	2	1.1
eSint16	3	1.1
DitherMatrix		
eDeviceBest	0	1.1
DuplexPageMode		
eDuplexHorizontalBinding	0	1.1
eDuplexVerticalBinding	1	1.1
DuplexPageSide		
eFrontMediaSide	0	1.1
eBackMediaSide	1	1.1
ErrorReport		
eNoReporting	0	1.1
eBackChannel(BackCh)	1	1.1
eErrorPage(ErrPage)	2	1.1
eBackChAndErrPage	3	1.1
eNWBackChannel	4	2.0
eNWErrorPage	5	2.0
eNWBackChAndErrPage	6	2.0
FillMode		
eNonZeroWinding	0	1.1
eEvenOdd	1	1.1
LineCapStyle		
eButtCap	0	1.1
eRoundCap	1	1.1
eSquareCap	2	1.1
eTriangleCap	3	1.1
LineJoin		
eMiterJoin	0	1.1
eRoundJoin	1	1.1
eBevelJoin	2	1.1
eNoJoin	3	1.1
Measure		
eInch	0	1.1
eMillimeter	1	1.1
eTenthsOfAMillimeter	2	1.1
MediaSize Enumerations		
eDefaultPaperSize	96	2.1
eLetterPaper	0	1.1
eLegalPaper	1	1.1

eA4Paper	2	1.1
eExecPaper	3	1.1
eLedgerPaper	4	1.1
eA3Paper	5	1.1
eCOM10Envelope	6	1.1
eMonarchEnvelope	7	1.1
eC5Envelope	8	1.1
eDLEnvelope	9	1.1
eJB4Paper	10	1.1
eJB5Paper	11	1.1
eB5Paper	13	2.1
eB5Envelope	12	1.1
eJPostcard	14	1.1
eJDoublePostcard	15	1.1
eA5Paper	16	1.1
eA6Paper	17	2.0
eJB6Paper	18	2.0
JIS8K	19	2.1
JIS16K	20	2.1
JISExec	21	2.1
MediaSource		
eDefaultSource	0	1.1
eAutoSelect	1	1.1
eManualFeed	2	1.1
eMultiPurposeTray	3	1.1
eUpperCassette	4	1.1
eLowerCassette	5	1.1
eEnvelopeTray	6	1.1
eThirdCassette	7	2.0
External Trays* 1-248	8-255	2.0
MediaDestination		
eDefaultDestination	0	2.0
eFaceDownBin	1	2.0
eFaceUpBin	2	2.0
eJobOffsetBin	3	2.0
External Bins** 1-251	5-255	
Orientation		
ePortraitOrientation	0	1.1
eLandscapeOrientation	1	1.1
eReversePortrait	2	1.1
eReverseLandscape	3	1.1
eDefaultOrientation	4	2.1
PatternPersistence		
eTempPattern	0	1.1
ePagePattern	1	1.1
eSessionPattern	2	1.1
SymbolSet		
See Appendix O.		

SimplexPageMode		
eSimplexFrontSide	0	1.1
TxMode		
eOpaque	0	1.1
eTransparent	1	1.1
WritingMode		
eHorizontal	0	2.0
eVertical	1	2.0

* External input trays 1 through 248 are selected by substituting the values 8 through 255 for the enumerated values. Example, 8 = first external input tray, 9 = second external input tray, etc.

** External output bins 1 through 251 are selected by substituting the values 5 through 255 for the enumerated values. Example, 5 = first external output bin, 6 = second external output bin, etc.

Appendix K. PCL XL Error and Warning Codes

Generic Operator Errors

PCL XL reports several error codes that apply to more than one operator in a general way. Generic operator error codes are described in this section.

Generic Errors

IllegalOperatorSequence	PCL XL read an operator that is out-of-sequence according to the legal sequencing of operators defined by the protocol. For example, a *ReadImage* occurring prior to a *BeginImage* yields this error.
IllegalTag	PCL XL had expected to read a data tag or operator tag and instead read something that is undefined for the current version of PCL XL.
InsufficientMemory	The amount of memory required to complete the current operation is unavailable.
InternalOverflow	Completing the current operation requires an amount of one or more internal resources that exceeds the maximum allowed.

Generic Attribute Errors

IllegalArraySize	PCL XL may yield this error for any array that cannot be zero through infinity in length (see the Attribute ID table in the appendix for valid array sizes).
IllegalAttribute	An attribute provided to an operator is not valid for that operator. For example, the *BeginPage* operator given a *Point* attribute yields this error.
IllegalAttributeCombination	An operator has two or more attributes that either conflict or do not make sense when presented together. For example, providing *BeginPage* with both the *MediaSize* and *CustomMediaSize* attribute yields this error.
IllegalAttributeDataType	The data type of the data value provided to an attribute is not valid for that attribute. For example, the single-value "*sint16"* data type is not valid for the Point attribute which requires two values: one each for the x and y coordinates of the point. The "*sint16_xy*" data type is valid for the Point attribute.
IllegalAttributeValue	PCL XL will yield this error if an attribute value given is out-of-range for the attribute value expected.
MissingAttribute	An operator with required attributes is missing one or more of the required attributes. For example, *SetCursor* without the *Point* attribute yields this error.

Generic Cursor Errors

CurrentCursorUndefined	An operator needing the current cursor position in the current path found that there was no current path and therefore no current cursor position. For example, *SetCursorRel* yields this error if performed immediately following a *NewPath* operator.

Generic Font Errors

NoCurrentFont	An operator needing a valid current font found that no font is set in the current graphics state. For example, attempting to perform the *Text or TextPath* operators without a successful *SetFont* operation for the current graphics state yields this error.
BadFontData	The font and/or character description data for the current font operation is incompatible with the current font operation. For example, attempting to perform character scaling on a bitmap characters by setting CharScale to something other

	than 1.0 will yield this error.

Generic Data Source Errors

DataSourceNotOpen	An operator that requires data from the data source yields this error when no data source is open.
ExtraData	An operator reading the *default* data source and finding that there is more data than required for the operation yields this error.
IllegalDataLength	An operator reading the *default* data source and finding that the length of data is not either a required constant length nor a required multiple of a constant length yields this error. For example, the BezierPath operator yields this error if the length of data in the default data source is not a multiple of 3.
IllegalDataValue	An operator that reads the data source and finds that a field in the data source contains an illegal value for that operator yields this error. For example, an illegal value in the x-pair type field for the *ScanLineRel* operator yields this error.
MissingData	An operator reading the data source and finding that there is less data than is required to complete the operation yields this error.

Operator-Specific Errors

PCL XL reports several error codes that are specific to an operator. Specific operator errors are described in this section.

BeginChar Errors

CannotReplaceCharacter	An attempt was made to download a character to an internal font or to a font on a mass-storage device.
FontUndefined	The user is attempting to start a character definition with *BeginChar*, but the font name specified is undefined.

BeginFontHeader Errors

FontNameAlreadyExists	The font name given already exists. The font must be removed before re-defining the header.

BeginImage/BeginRastPattern Errors

ImagePaletteMismatch	The palette is too small or too large for the ColorDepth specified for the indexed image or pattern.
MissingPalette	There is no palette for the indexed image/pattern.

BeginPage Warnings

IllegalMediaSize	The values for MediaSize or CustomMediaSize or CustomMediaSizeUnits are out of range for the current device.
IllegalMediaSource	The value for MediaSource is out of range for the current device or device configuration.
IllegalMediaDestination	The value for MediaDestination is out of range for the current device or device configuration.
IllegalOrientation	The value for page orientation is out of range for the current device.

OpenDataSource Errors

DataSourceNotClosed	A data source is already open (not closed).

PushGS Errors

MaxGSLevelsExceeded	The maximum number of graphics state save levels has been exceeded. The maximum level is determined by the available memory.

ReadChar Errors

FSTMismatch	The value of the character format field is inconsistent with the value of the Font Scaling Technology field in the corresponding font header.
UnsupportedCharacterClass	The value of the character class field represents a character class that is not supported by the current version of PCL XL.
UnsupportedCharacterFormat	The value of the character format field represents a character format that is not supported by the current version of PCL XL.
IllegalCharacterData	The character data is in a format unrecognized by PCL XL.

ReadFontHeader Errors

IllegalFontData	The font header data is in a format unrecognized by PCL XL.
IllegalFontHeaderFields	The font header Font Format, Font Scaling Technology, Variety, and/or Orientation fields contain invalid values.
IllegalNullSegmentSize	The size of the Null segment in the font header is invalid.
IllegalFontSegment	A segment included in the font header is not defined for the corresponding font scaling technology.
MissingRequiredSegment	A segment required by the corresponding font scaling technology is missing.
IllegalGlobalTrueTypeSegment	The global TrueType segment is in a format unrecognized by PCL XL.
IllegalGalleyCharacterSegment	The galley character segment is in a format unrecognized by PCL XL.
IllegalVerticalTxSegment	The vertical transformation segment is in a format unrecognized by PCL XL.
IllegalBitmapResolutionSegment	The bitmap resolution segment is in a format unrecognized by PCL XL.

RemoveFont Warnings

UndefinedFontNotRemoved	An attempt was made to remove an undefined font.
InternalFontNotRemoved	An attempt was made to remove an internal font.
MassStorageFontNot Removed	An attempt was made to remove a font on a mass storage device.

ExecStream Errors

StreamUndefined	An attempt was made to execute an undefined stream.
IllegalOpSequence	An attempt was made to execute a stream within another stream (Stream Nesting) when the protocol class is set to less than 2.1.
StreamNestingError	An attempt was made by a stream to start another copy of itself.
StreamAlreadyRunning	An attempt was made to start another copy of an already executing stream.
StreamStackFull	An attempt was made to start more that 32 nested streams.
InternalStreamError	Normally XL cannot exit and clean up while streams are executing. If this does occur and XL cannot safely clean up the stream stack entries, this error will result.

RemoveStream Warnings

UndefinedStreamNot Removed	An attempt was made to remove an undefined stream.
InternalStreamNotRemoved	An attempt was made to remove an internal stream.
MassStorageStreamNotRemoved	An attempt was made to remove a stream on a mass storage device.
StreamAlreadyRunning	An attempt was made to remove a copy of an already executing stream.

SetBrushSource/SetPenSource Errors

ColorSpaceMismatch	The color attribute for SetPen/SetBrush (i.e. RGBColor or GrayLevel) does not match the ColorSpace at the current graphics state level. The ColorSpace was either set by default (at BeginPage) or by the most recent SetColorSpace operator.
RasterPatternUndefined	The value given for PatternSelectID does not identify a raster pattern.

SetClipReplace/SetClipIntersect/SetClipRectangle Errors

ClipModeMismatch	The ClipRegion attribute is “eExterior” and the ClipMode is “eNonZeroWinding.” When the ClipRegion is “eExterior” the ClipMode must be “eEvenOdd.”

SetFont Errors

FontUndefinedNoSubstituteFound	An attempt was made to set a font that was not defined in the device. A font substitution could not be found by PCL XL.
SymbolSetRemapUndefined	The font is defined, but unusable with the requested symbol set.

SetFont Warnings

“font name” substituted for “font name”	An attempt was made to set a font that was undefined in the device. A substitution was found and made by PCL XL.

Stream Header Errors

UnsupportedBinding	PCL XL does not support the binding requested in the stream header.
UnsupportedClassName	The class name is unknown to PCL XL (e.g. “PCL-XY”).
UnsupportedProtocol	The protocol class number and/or revision is not supported by current version of PCL XL.
IllegalStreamHeader	The stream header is in a format unrecognized by PCL XL.

Appendix Q. Compression Methods

The PCL XL **BeginImage** and **BeginRastPattern** operators allow the user to specify compression methods for the image and pattern data. A form of run-length encoding compression is supported for protocol class 1.1 of PCL XL. JPEG compression has been added to protocol class 2.0 of PCL XL.

PCL XL Run Length Encoding Compression Method (eRLECompression)

The PCL XL RLE compression method employs control bytes followed by data bytes. Each control byte in the compressed data sequence is a signed, two's complement byte.

If bit 7 of the control byte is zero (0 <= control byte <= 127) the bytes following are *literal*. Literal bytes are simply uncompressed data bytes. The number of literal bytes following a control byte is one plus the value of the control byte. Thus, a control byte of 0 means 1 literal byte follows; a control byte of 6 means 7 literal bytes follow; and so on.

If bit 7 of the control byte is 1 (-127 <= control byte <= -1), the byte following the control byte will occur two or more times as decompressed data. A byte following a control byte in this range is called a *repeat* byte. The control byte's absolute value plus one is the number of times the byte following will occur in the decompressed sequence of bytes. For example, a control byte of -5 means the subsequent byte will occur 6 times as decompressed data.

A control byte of -128 is ignored and is not included in the decompressed data. The byte following a control byte of 128 is treated as the next control byte.

It is more efficient to code two consecutive identical bytes as a repeated byte, unless these bytes are preceded and followed by literal bytes. Three-byte repeats should always be encoded using a repeat control byte.

The following is an example of PCL XL RLE compression encoding where tokens within single quotes are single byte character values and the numeric tokens are single byte numbers.

Compressed Bytes	-5 'I' 0 'S' -1 'E'
Uncompressed Bytes	'I' 'I' 'I' 'I' 'I' 'I' 'S' 'E' 'E'

PCL XL JPEG Compression Method (eJPEGCompression).

JPEG is an acronym for **"Joint PhotographicExperts Group."** This groups has been working under the auspices of three major international standards organizations – the **ISO, CCITT and IEC** – for the purpose of developing a standard for color image compression. This JPEG software is based in part on the work of the Independent JPEG Group.

The JPEG implementation in PCL XL protocol class 2.0 decodes gray-scale or color image data in JPEG baseline encoded format. All information required for decoding

the image is contained within the JPEG signaling parameters, which accompany the encoded data in the compressed data stream.

PCL XL Delta Row Compression Method (eDeltaRowCompression)

The PCL XL implementation of Delta Row compression attempts to follow the PCL 5 Implementor's Guide definition, specified in Chapter 13.

In each delta row, there is a command byte and replacement bytes. The command byte has 2 parts, where bits 0 through 4 are the Offset from the current byte and bits 5 through 7 are the number of replacement bytes.

The PCL XL implementation follows the PCL5 implementation except in the following:

1) the seed row is initialized to zeroes and contains the number of bytes defined by SourceWidth in the BeginImage operator.

2) the delta row is preceded by a 2-byte byte count which indicates the number of bytes to follow for the delta row. The byte count is expected to be in LSB MSB order.

3) to repeat the last row, use the 2-byte byte count of 00 00.

The control byte has the following format:

Number of delta bytes: Bits 5-7 indicate the number of consecutive replacement bytes that follow the commands byte. The actual number of of replacement bytes is always one more than the value (000 = 1, 111 = 8). If more than 8 delta bytes are needed, additional "command byte/delta bytes" are added.

[(Command Byte) (1-8 Delta Bytes)] [(Command Byte) (1-8 Delta Bytes)] . . .

Offset: Bits 0-4 show where to position the replacement byte string. This is the offset: it specifies a byte placement, counting from left to right from the current byte position. The "current" byte is the first unaltered byte that follows the last replacement bytes; at the the beginning of a row, the current byte immediately follows the left raster margin. Bits 0-4 allow a maximum value of 31, but larger offsets are possible. A value of 0 to 30 indicates the delta bytes are offset from the 1st to the 31st bytes. A value of 31 indicates that an additional offset byte follows the command byte.

To summarize, bits 0-4 have the following meaning:

0 to 30: the offset is 0 to 30.

31: the offset is 31 or greater. If the offset is 31, an additional offset byte follows the command byte. The offset in the command bytes is added to the offset bytes. If the offset byte is 0, the offset is 31; if the offset byte is 255 additional offset bytes follow. The last offset byte will have a value less than 255. All the offset bytes are added to the offset in the command byte to get the offset value. For example, if there are two offset bytes, and the last byte contains 175, the total offset would be: 31+255+175=461.

The following example shows the hexadecimal values of RGB 8Bit Direct compressed data using Delta Row encoding:

```
ubyte eRGB ColorSpace
SetColorSpace

ubyte eDirectPixel ColorMapping
ubyte e8Bit ColorDepth
uint16 32 SourceWidth
uint16 32 SourceHeight
uint16_xy 237 237 DestinationSize
BeginImage

uint16 0 StartLine
uint16 32 BlockHeight
ubyte eDeltaRowCompression CompressMode
ReadImage

dataLengthByte 192
hex_raw* [
40 00       // 64 (decimal) bytes follow
43 FF FF FF 43 FF FF FF 43 FF FF FF 43 FF FF FF
            // 43= 010 00011 = 2+1 bytes follow with 3 bytes offset
43 FF FF FF 43 FF FF FF 43 FF FF FF 43 FF FF FF
43 FF FF FF 43 FF FF FF 43 FF FF FF 43 FF FF FF
43 FF FF FF 43 FF FF FF 43 FF FF FF 43 FF FF FF
00 00       // repeat the last row
. . .

This data would expand as follows:
First Row:
00 00 00 FF FF FF 00 00 00 FF FF FF 00 00 00 FF FF FF 00 00 00 FF FF FF
00 00 00 FF FF FF 00 00 00 FF FF FF
00 00 00 FF FF FF 00 00 00 FF FF FF 00 00 00 FF FF FF 00 00 00 FF FF FF
00 00 00 FF FF FF 00 00 00 FF FF FF
00 00 00 FF FF FF 00 00 00 FF FF FF 00 00 00 FF FF FF 00 00 00 FF FF FF
Second Row: (repeat last row)
```