



*International  
Color Consortium*®

---

**Specification  
ICC.1:2001-04**

*File Format for Color Profiles*

---

**[REVISION of ICC.1:1998-09]**

## Copyright notice

Copyright © 1994-2001 International Color Consortium®

Permission is hereby granted, free of charge, to any person obtaining a copy of the Specification and associated documentation files (the "Specification") to deal in the Specification without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Specification, and to permit persons to whom the Specification is furnished to do so, subject to the following conditions.

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Specification.

The Specification is provided "as is", without warranty of any kind, express, implied, or otherwise, including but not limited to the warranties of merchantability, fitness for a particular purpose and noninfringement. In no event shall the International Color Consortium be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of, or in connection with the Specification or the use or other dealings in the Specification.

Except as contained in this notice, the name of the International Color Consortium shall not be used in advertising or otherwise to promote the use or other dealings in this Specification without prior written authorization from the International Color Consortium.

## Licenses and trademarks

International Color Consortium and the ICC logo are registered trademarks of the International Color Consortium.

Rather than put a trademark symbol in every occurrence of other trademarked names, we state that we are using the names only in an editorial fashion, and to the benefit of the trademark owner, with no intention of infringement of the trademark.

## For additional information on the ICC

Visit the ICC Web site: <http://www.color.org>

## Founding Members

Adobe Systems Inc.  
AGFA  
Apple Computer, Inc.  
Eastman Kodak Company  
Microsoft Corporation  
Silicon Graphics, Inc.  
Sun Microsystems, Inc.

## Regular Members

Acer Peripherals, Inc.  
ALWAN COLOR Expertise  
B & R, groupe ODESSA  
Barco N.V.  
Binuscan, Inc.  
Canon R & D Center Americas, Inc.  
Color Savvy Systems, Inc.  
Computer and Software and Technology  
Corbis Corporation  
CreoScitex  
Dainippon Screen  
DuPont Ink Jet  
Fuji Photo Group  
Fuji Xerox Co., Ltd.  
Fujitsu Laboratories Ltd.  
Gretag-Macbeth  
Harlequin Group plc.  
Heidelberger Druckmaschinen AG.  
Hewlett Packard Company  
Imaging Business Machines, LLC  
Imaging Technologies Corp. (ITEC)  
Imaging, S.A.  
Imation  
Industrial Technology Research Institute (ITRI)  
Intel Corporation  
Just Normlicht  
Konica Corporation  
Korea C4 Co., Ltd.  
Kyocera Corp.  
Lexmark International  
Lotsadots, Inc.  
Matsushita Electric Industrial Co., Ltd.  
Minolta Co., Ltd.  
Minolta-QMS, Inc.  
Miro Displays  
NEC Corporation

Nikon Corp.  
Oak Technology, Inc.  
Okidata  
Onyx Graphics Corp.  
Pantone, Inc.  
Polaroid Corporation  
Quark, Inc.  
QUATOGRAPHIC AG  
Que-Net Media  
R.R. Donnelley & Sons, Inc  
R.T. Image  
RATIO Entwicklungen GmbH  
Richo Corporation  
Royal Information Electronics Company, Ltd.  
SCP Software GmbH  
Seiko Epson Corp.  
SELECTA  
Sharp Laboratories of America, Inc.  
Shira Inc.  
Sony Corporation  
Toppan Printing Co., Ltd.  
Toyo Ink Mfg. Co., Ltd.  
Vidar Systems Corp.  
WaveMark Technologies, Inc.  
WayTech Development, Inc.  
Xerox Corporation  
X-Rite

## Honorary Members

EPFL  
FOGRA  
London College of Printing

Note: This list is correct at time of publication. See the ICC Web Site for the most up-to-date member list.

## Contents

0	Introduction.....	1
0.1	Intended audience .....	1
0.2	Organizational description of this specification .....	1
0.3	International Color Consortium .....	2
0.4	Device profiles .....	2
0.5	Profile element structure .....	3
0.6	Embedded profiles .....	3
0.7	Registration authority .....	3
0.8	Redundant data arbitration .....	3
1	Scope .....	4
2	Normative references .....	4
3	Conformance .....	5
4	Definitions.....	5
5	Notation, symbols and abbreviations.....	6
5.1	Notations.....	6
5.2	Symbols and abbreviations.....	6
5.3	Basic numeric types.....	7
5.3.1	dateTimeNumber.....	7
5.3.2	response16Number.....	7
5.3.3	s15Fixed16Number.....	8
5.3.4	u16Fixed16Number.....	8
5.3.5	u8Fixed8Number.....	8
5.3.6	uint16Number.....	8
5.3.7	uint32Number.....	8
5.3.8	uint64Number.....	9
5.3.9	uint8Number.....	9
5.3.10	XYZNumber.....	9
5.3.11	Seven-bit ASCII.....	10
6	Requirements .....	11
6.1	Header description .....	13
6.1.1	Profile size.....	13
6.1.2	CMM Type signature .....	14
6.1.3	Profile version.....	14
6.1.4	Profile/Device Class signature .....	14
6.1.5	Color Space signature.....	15
6.1.6	Profile Connection Space signature .....	16
6.1.7	Primary Platform signature .....	16
6.1.8	Profile flags.....	17
6.1.9	Device manufacturer and model signatures.....	17
6.1.10	Attributes .....	17
6.1.11	Rendering intent .....	17
6.1.12	Profile Creator signature .....	18
6.2	Tag table definition.....	18
6.2.1	Tag signature .....	18
6.2.2	Offset.....	18

6.2.3	Element size .....	19
6.3	Required tags for profiles.....	19
6.3.1	Input Profile .....	20
6.3.1.1	Monochrome Input Profiles .....	20
6.3.1.2	Three-component Matrix-based Input Profiles.....	21
6.3.1.3	N-component LUT-based Input Profiles .....	22
6.3.2	Display Profile .....	23
6.3.2.1	Monochrome Display Profiles .....	23
6.3.2.2	RGB Display Profiles .....	24
6.3.3	Output Profile .....	25
6.3.3.1	Monochrome Output Profiles .....	25
6.3.3.2	Color Output Profiles .....	26
6.3.4	Additional Profile Formats .....	27
6.3.4.1	Device Link Profile .....	27
6.3.4.2	ColorSpace Conversion Profile.....	27
6.3.4.3	Abstract Profile .....	28
6.3.4.4	Named Color Profile .....	28
6.4	Tag descriptions.....	29
6.4.1	AToB0Tag .....	31
6.4.2	AToB1Tag .....	31
6.4.3	AToB2Tag .....	31
6.4.4	blueColorantTag.....	31
6.4.5	blueTRCTag .....	31
6.4.6	BToA0Tag .....	31
6.4.7	BToA1Tag .....	32
6.4.8	BToA2Tag .....	32
6.4.9	calibrationDateTimeTag .....	32
6.4.10	charTargetTag.....	32
6.4.11	chromaticAdaptationTag .....	32
6.4.12	chromaticityTag .....	33
6.4.13	copyrightTag.....	33
6.4.14	crdInfoTag .....	33
6.4.15	deviceMfgDescTag.....	33
6.4.16	deviceModelDescTag.....	34
6.4.17	deviceSettingsTag.....	34
6.4.18	gamutTag .....	34
6.4.19	grayTRCTag.....	34
6.4.20	greenColorantTag .....	35
6.4.21	greenTRCTag.....	35
6.4.22	luminanceTag.....	35
6.4.23	measurementTag .....	35
6.4.24	mediaBlackPointTag .....	35
6.4.25	mediaWhitePointTag .....	35
6.4.26	namedColorTag.....	35
6.4.27	namedColor2Tag.....	36
6.4.28	outputResponseTag .....	36
6.4.29	preview0Tag.....	36
6.4.30	preview1Tag.....	36
6.4.31	preview2Tag.....	36
6.4.32	profileDescriptionTag .....	36
6.4.33	profileSequenceDescTag .....	37
6.4.34	ps2CRD0Tag.....	37
6.4.35	ps2CRD1Tag.....	37

6.4.36	ps2CRD2Tag.....	37
6.4.37	ps2CRD3Tag.....	37
6.4.38	ps2CSATag.....	38
6.4.39	ps2RenderingIntentTag.....	38
6.4.40	redColorantTag .....	38
6.4.41	redTRCTag.....	38
6.4.42	screeningDescTag .....	38
6.4.43	screeningTag.....	39
6.4.44	technologyTag.....	39
6.4.45	ucrbgTag .....	39
6.4.46	viewingCondDescTag .....	40
6.4.47	viewingConditionsTag .....	40
6.5	Tag type definitions.....	40
6.5.1	chromaticityType .....	41
6.5.2	crdInfoType .....	42
6.5.3	curveType.....	43
6.5.4	dataType .....	43
6.5.5	dateTimeType .....	44
6.5.6	deviceSettingsType .....	44
6.5.7	lut16Type.....	46
6.5.8	lut8Type.....	49
6.5.9	measurementType .....	52
6.5.10	namedColorType.....	54
6.5.11	namedColor2Type.....	54
6.5.12	profileSequenceDescType .....	55
6.5.13	responseCurveSet16Type.....	56
6.5.14	s15Fixed16ArrayType .....	59
6.5.15	screeningType.....	59
6.5.16	signatureType.....	60
6.5.17	textDescriptionType.....	60
6.5.18	textType.....	62
6.5.19	u16Fixed16ArrayType .....	63
6.5.20	ucrbgType .....	63
6.5.21	ulnt16ArrayType.....	64
6.5.22	ulnt32ArrayType.....	64
6.5.23	ulnt64ArrayType.....	64
6.5.24	ulnt8ArrayType.....	65
6.5.25	viewingConditionsType .....	65
6.5.26	XYZType .....	65
Annex A		
Color spaces.....		66
A.1	Profile Connection Spaces .....	66
A.2	External and internal conversions .....	70
A.3	Rendering Intents .....	70
A.3.1	Colorimetric Intents.....	70
A.3.1.1	MediaWhitePoint Tag.....	70
A.3.1.2	Media-Relative Colorimetric Intent .....	71
A.3.1.3	ICC-Absolute Colorimetric Intent.....	71
A.3.1.4	Applying the ICC-Absolute Colorimetric Intent.....	71
A.3.2	Perceptual Intent .....	71
A.3.3	Saturation Intent .....	71

Annex B	
Embedding Profiles .....	72
B.1 Embedding ICC profiles in PICT files .....	72
B.2 Embedding ICC profiles in EPS files .....	73
B.3 Embedding ICC profiles in TIFF files .....	75
B.4 Embedding ICC profiles in JFIF files .....	75
B.5 Embedding ICC profiles in GIF files .....	76
Annex C	
PostScript Level 2 Tags.....	77
C.1 Synchronizing profiles and CRDs.....	77
Annex D	
Profile Connection Space explanation.....	80
D.1 Introduction.....	80
D.2 Colorimetry and its interpretation .....	81
D.3 Color measurements .....	82
D.4 Colorimetry corrections and adjustments in Output Profiles .....	83
D.5 Output to reflection print media .....	83
D.6 Output to transparency media.....	84
D.7 Negative media .....	85
D.8 Monitor display .....	85
D.9 Colorimetry corrections and adjustments in Input Profiles .....	85
D.10 Scanned reflection prints.....	85
D.11 Scanned transparencies.....	86
D.12 Scanned negatives.....	86
D.13 Computer graphics.....	87
D.14 Scene capture .....	87
D.15 Colorimetric input .....	87
D.16 Techniques for colorimetry corrections .....	88
Annex E	
Chromatic Adaptation Tag.....	89
E.1 Calculating the Chromatic Adaptation Matrix .....	89
E.1.1 von Kries transformation .....	89
E.1.2 Linearized Bradford/CIECAM97 transformation .....	90
E.1.3 “Wrong” von Kries transformation. ....	91
E.2 Applying the Chromatic Adaptation Matrix .....	91
Annex F	
Summary of spec changes .....	93

**Tables**

Table 1 — dateTimeNumber .....	7
Table 2 — response16Number .....	7
Table 3 — s15Fixed16Number .....	8
Table 4 — u16Fixed16Number .....	8
Table 5 — u8Fixed8Number .....	8
Table 6 — XYZNumber .....	9
Table 7 — Hexadecimal .....	9
Table 8 — Decimal .....	10
Table 9 — Header .....	12
Table 10 — Profile version .....	13
Table 11 — Device class .....	13
Table 12 — Profile class .....	14
Table 13 — Color space signature .....	14
Table 14 — Profile connection space signature .....	15
Table 15 — Primary platform signature .....	15
Table 16 — Profile flags .....	16
Table 17 — Header attributes .....	16
Table 18 — Header rendering intents .....	17
Table 19 — Tag table structure .....	17
Table 20 — Profile type/profile tag and defined rendering intents .....	19
Table 21 — Monochrome input profile required tags .....	19
Table 22 — Three-component matrix-based input profile required tags .....	20
Table 23 — N-component LUT-based input profile required tags .....	21
Table 24 — Monochrome display profile required tags .....	22
Table 25 — RGB display profile required tags .....	23
Table 26 — Monochrome output profile required tags .....	24
Table 27 — Color output profile required tags .....	25
Table 28 — Device link profile required tags .....	25
Table 29 — ColorSpace conversion profile required tags .....	26
Table 30 — Abstract profile required tags .....	27
Table 31 — Named color required tags .....	27
Table 32 — Tag list .....	28
Table 33 — Technology signatures .....	37
Table 34 — chromaticityType encoding .....	39
Table 35 — Phosphor or colorant encoding .....	40
Table 36 — crdInfoType encoding .....	40
Table 37 — curveType encoding .....	41
Table 38 — dataType encoding .....	41
Table 39 — dateTimeType encoding .....	42
Table 40 — deviceSettingsType encoding .....	42
Table 41 — Platform encoding .....	42
Table 42 — Device settings combinations structure .....	43
Table 43 — Device settings setting structure .....	43
Table 44 — Device settings ID signatures .....	43
Table 45 — Device settings media type encoding .....	44



Table 46 — Device settings halftone encoding .....	44
Table 47 — lut16Type encoding .....	45
Table 48 — lut16Type channel encodings .....	47
Table 49 — lut8Type encoding .....	48
Table 50 — lut8Type channel encodings .....	50
Table 51 — measurementType structure .....	50
Table 52 — Standard observer encodings .....	51
Table 53 — Measurement geometry encodings .....	51
Table 54 — Measurement flare encodings .....	51
Table 55 — Standard illuminant encodings .....	51
Table 56 — namedColorType encoding .....	52
Table 57 — namedColor2Type encoding .....	53
Table 58 — profileSequenceDescType structure .....	54
Table 59 — Profile Description structure .....	54
Table 60 — responseCurveSet16Type structure .....	55
Table 61 — Curve structure .....	56
Table 62 — Curve measurement encodings .....	56
Table 63 — s16Fixed16ArrayType encoding .....	57
Table 64 — screeningType structure .....	57
Table 65 — Screening flag encodings .....	57
Table 66 — Screening spot encodings .....	58
Table 67 — signatureType encoding .....	58
Table 68 — textDescriptionType encoding .....	59
Table 69 — textDescriptionType encoding example .....	60
Table 70 — textType encoding .....	60
Table 71 — u16Fixed16ArrayType encoding .....	61
Table 72 — ucrbgType structure .....	61
Table 73 — uInt16ArrayType encoding .....	62
Table 74 — uInt32ArrayType encoding .....	62
Table 75 — uInt64ArrayType encoding .....	62
Table 76 — uInt8ArrayType encoding .....	63
Table 77 — viewingConditionsType encoding .....	63
Table 78 — XYZType encoding .....	63
Table 79 — CIE color spaces .....	64
Table 80 — CIEXYZ encoding .....	66
Table 81 — CIELAB L* encoding .....	67
Table 82 — CIELAB a* or b* encoding .....	67
Table 83 — PICT selectors .....	70
Table 84 — ICC profile IFD entry structure .....	73

## 0 Introduction

This specification describes the International Color Consortium® profile format. The intent of this format is to provide a cross-platform device profile format. Such device profiles can be used to translate color data created on one device into another device's native color space. The acceptance of this format by operating system vendors allows end users to transparently move profiles and images with embedded profiles between different operating systems. For example, this allows a printer manufacturer to create a single profile for multiple operating systems.

A large number of companies and individuals from a variety of industries participated in very extensive discussions on these issues. Many of these discussions occurred under the auspices of Forschungsgesellschaft Druck e.V. (FOGRA), the German graphic arts research institute, during 1993. The present specification evolved from these discussions and the ColorSync™ 1.0 profile format.

This is a very complex set of issues and the organization of this document strives to provide a clear, clean, and unambiguous explanation of the entire format. To accomplish this, the overall presentation is from a top-down perspective, beginning with a summary overview and continuing down into more detailed specifications to a byte stream description of format.

### 0.1 Intended audience

This specification is designed to provide developers and other interested parties a clear description of the profile format. A nominal understanding of color science is assumed, such as familiarity with the CIELAB color space, general knowledge of device characterizations, and familiarity of at least one operating system level color management system.

### 0.2 Organizational description of this specification

This specification is organized into a number of major clauses and annexes. Each clause and subclause is numbered for easy reference. A brief introduction is followed by a detailed summary of the issues involved in this document including: International Color Consortium, device profiles, profile element structure, embedded profiles, registration authority, and color model arbitration.

Clause 1 describes the scope of this specification.

Clause 2 provides the normative references for this specification.

Clause 3 describes the conformance requirements for this specification.

Clause 4 provides general definitions used within this specification.

Clause 5 provides descriptions of notations, symbols and abbreviations used in this specification.

Clause 6 describes the requirements of this specification. 6.1 describes the format header definition. 6.2 describes the tag table. 6.3 provides a top level view of what tags are required for each type of profile classification and a brief description of the algorithmic models associated with these classes. Four additional color transformation formats are also described: device link, color space conversion, abstract transformations, and named color transforms. 6.4 is a detailed algorithmic and intent description of all of the tagged elements described in the previous clauses. 6.5 provides a byte stream definition of the structures that make up the tags in 6.4.

Annex A describes the color spaces used in this specification. Annex B provides the necessary details to embed profiles into PICT, EPS, TIFF, and JFIF files. Annex C provides a general description of the Post-

Script Level 2 tags used in this specification. Annex D is a paper describing details of the profile connection space. Annex E provide details on the chromaticAdaptationTag. Annex F is a summary of the changes made in the last few revisions of the spec. The C language ICC header file has been removed as an appendix. It is available on the ICC web site as file ICC.3.

### 0.3 International Color Consortium

Considering the potential impact of this specification on various industries, a consortium has been formed that will administer this specification and the registration of tag signatures and descriptions. The founding members of this consortium are: Adobe Systems Inc., Agfa-Gevaert N.V., Apple Computer, Inc., Eastman Kodak Company, FOGRA (Honorary), Microsoft Corporation, Silicon Graphics, Inc., Sun Microsystems, Inc., and Taligent, Inc (resigned). These companies have committed to fully support this specification in their operating systems, platforms and applications.

### 0.4 Device profiles

Device profiles provide color management systems with the information necessary to convert color data between native device color spaces and device independent color spaces. This specification divides color devices into three broad classifications: input devices, display devices and output devices. For each device class, a series of base algorithmic models are described which perform the transformation between color spaces. These models provide a range of color quality and performance results. Each of the base models provides different trade-offs in memory footprint, performance and image quality. The necessary parameter data to implement these models is described in the required portions on the appropriate device profile descriptions. This required data provides the information for the color management framework default color management module (CMM) to transform color information between native device color spaces. A representative architecture using these components is illustrated in Figure 1 below.

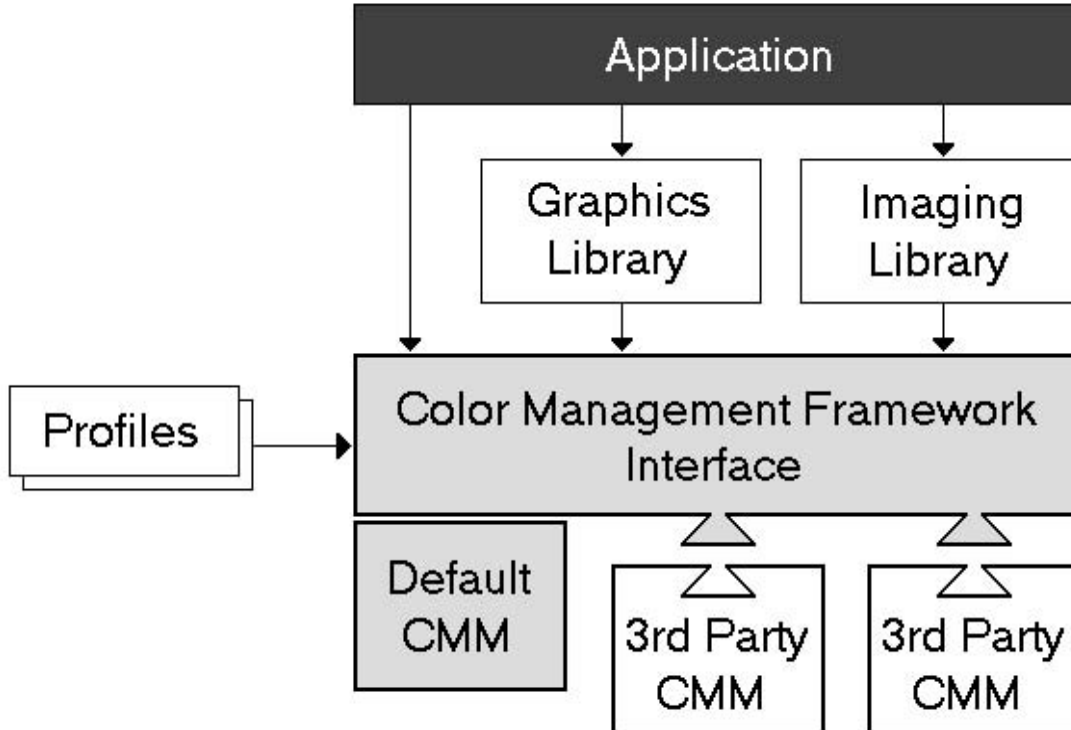


Figure 1 — Color management architecture

## 0.5 Profile element structure

The profile structure is defined as a header followed by a tag table followed by a series of tagged elements that can be accessed randomly and individually. This collection of tagged elements provides three levels of information for developers: required data, optional data and private data. An element tag table provides a table of contents for the tagging information in each individual profile. This table includes a tag signature, the beginning address offset and size of the data for each individual tagged element. Signatures in this specification are defined as a four-byte hexadecimal number. This tagging scheme allows developers to read in the element tag table and then randomly access and load into memory only the information necessary to their particular software application. Since some instances of profiles can be quite large, this provides significant savings in performance and memory. The detailed descriptions of the tags, along with their intent, are included later in this specification.

The required tags provide the complete set of information necessary for the default CMM to translate color information between the profile connection space and the native device space. Each profile class determines which combination of tags is required. For example, a multi-dimensional lookup table is required for output devices, but not for display devices.

In addition to the required tags for each device profile, a number of optional tags are defined that can be used for enhanced color transformations. Examples of these tags include PostScript Level 2 support, calibration support, and others. In the case of required and optional tags, all of the signatures, an algorithmic description, and intent are registered with the International Color Consortium.

Private data tags allow CMM developers to add proprietary value to their profiles. By registering just the tag signature and tag type signature, developers are assured of maintaining their proprietary advantages while maintaining compatibility with this specification. However, the overall philosophy of this format is to maintain an open, cross-platform standard, therefore the use of private tags should be kept to an absolute minimum.

## 0.6 Embedded profiles

In addition to providing a cross-platform standard for the actual disk-based profile format, this specification also describes the convention for embedding these profiles within graphics documents and images. Embedded profiles allow users to transparently move color data between different computers, networks and even operating systems without having to worry if the necessary profiles are present on the destination systems. The intention of embedded profiles is to allow the interpretation of the associated color data. Embedding specifications are described in of this document.

## 0.7 Registration authority

This specification requires that signatures for CMM type, device manufacturer, device model, profile tags and profile tag types be registered to insure that all profile data is uniquely defined. The registration authority for these data is the ICC Technical Secretary. See the ICC Web Site ([www.color.org](http://www.color.org)) for contact information.

If and when this document becomes an International Standard this registration responsibility must be brought into conformance with ISO procedures. These procedures are being investigated on behalf of ICC and TC130.

## 0.8 Redundant data arbitration

There are several methods of color rendering described in the following structures that can function within a single CMM. If data for more than one method are included in the same profile, the following selection

algorithm should be used by the software implementation: if an 8-bit or 16-bit lookup table is present, it should be used; if a lookup table is not present (and not required), the appropriate default modeling parameters are used.

## 1 Scope

This specification defines the data necessary to describe the color characteristics used to input, display, or output images, and an associated file format for the exchange of this data.

## 2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this specification. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this specification are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of ISO and IEC maintain registers of currently valid International Standards.

CIE Publication 15.2-1986, "Colorimetry, Second Edition"

EBU Tech. 3213-E: EBU standard for chromaticity tolerances for studio monitors

ISO 5-1:1984, "Photography -- Density measurements -- Part 1: Terms, symbols and notations"

ISO 5-2:1991, "Photography -- Density measurements -- Part 2: Geometric conditions for transmission density"

ISO 5-4:1995, "Photography -- Density measurements -- Part 4: Geometric conditions for reflection density"

ISO/IEC 646:1991, "Information technology -- ISO 7-bit coded character set for information interchange"

ISO 3664:1975, "Photography -- Illumination conditions for viewing colour transparencies and their reproductions"

ISO/IEC 8824-1:1995, "Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation"

ISO/IEC 10918-1:1994, "Information technology -- Digital compression and coding of continuous-tone still images: Requirements and guidelines"

ISO/DIS 12234, "Photography -- Electronic still picture cameras -- Removeable memory (TIFF/EP)"

ISO/FDIS 12639, "Graphic Technology -- Prepress digital data exchange -- Tag image file format for image technology (TIFF/IT)"

ISO 12641:1997, "Graphic technology -- Prepress digital data exchange -- Colour targets for input scanner calibration"

ISO 12642:1996, "Graphic technology -- Prepress digital data exchange -- Input data for characterization of 4-colour process printing"

ISO 13655:1996, "Graphic technology -- Spectral measurement and colorimetric computation for graphic arts images"

ITU-R BT.709-2, Parameter values for the HDTV standards for production and international programme exchange

PICT Standard Specifications, published by Apple Computer, Inc.

PostScript Language Reference Manual, Second Edition, Adobe Systems Inc.

SMPTE RP 145-1994: SMPTE C Color Monitor Colorimetry

TIFF 6.0 Specification, published by Adobe Systems Incorporated.

### **3 Conformance**

Any color management system, application, utility or device driver that is in conformance with this specification shall have the ability to read the profiles as they are defined in this specification. Any profile-generating software and/or hardware that is in conformance with this specification shall have the ability to create profiles as they are defined in this specification. ICC conforming software will use the ICC profiles in an appropriate manner.

### **4 Definitions**

For the purposes of this specification, the following definitions shall apply:

#### **4.1**

##### **aligned**

A data element is aligned with respect to a data type if the address of the data element is an integral multiple of the number of bytes in the data type.

#### **4.2**

##### **ASCII string**

A sequence of bytes, each containing a graphic character from ISO/IEC 646, the last character in the string being a NULL (character 0/0).

#### **4.3**

##### **big-endian**

Addressing the bytes within a 16, 32 or 64-bit value from the most significant to the least significant, as the byte address increases.

#### **4.4**

##### **byte**

An 8-bit unsigned binary integer.

#### **4.5**

##### **byte offset**

The number of bytes from the beginning of a field.

#### **4.6**

##### **fixed point representation**

A method of encoding a real number into binary by putting an implied binary point at a fixed bit position. See Table 3 in 5.3.3 for an example.

Many of the tag types contain fixed point numbers. Several references can be found (MetaFonts, etc.) illustrating the preferability of fixed point representation to pure floating point representation in very structured circumstances.

**4.7****NULL**

The character coded in position 0/0 of ISO/IEC 646.

**4.8****profile connection space (PCS)**

An abstract color space used to connect the source and destination profiles. See A.1 for a full description.

**4.9****rendering intent**

A particular gamut mapping style or method of converting colors in one gamut to colors in another gamut. See Annex A for a more complete description of the rendering intents used in ICC profiles.

**5 Notation, symbols and abbreviations****5.1 Notations**

All numeric values in this specification are expressed in decimal, unless otherwise indicated. A letter “h” is suffixed to denote a hexadecimal value.

Literal strings are denoted in this specification by enclosing them in double quotation marks.

**5.2 Symbols and abbreviations**

The following symbols and abbreviations are used within this specification with the meanings indicated:

ANSI	American National Standards Institute
BG	Black Generation
CIE	<i>Commission Internationale de l'Éclairage</i> (International Commission on Illumination)
CLUT	Color Lookup Table (multidimensional)
CMM	Color Management Module
CMS	Color Management System
CMY	Cyan, Magenta, Yellow
CMYK	Cyan, Magenta, Yellow, Key (black)
CRD	Color Rendering Dictionary
CRT	Cathode-Ray Tube
EPS	Encapsulated PostScript
ICC	International Color Consortium
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
LCD	Liquid Crystal Display
LUT	Lookup Table
PCS	Profile Connection Space

PPD	PostScript Printer Description
RGB	Red, Green, Blue
TIFF	Tagged Image File Format
TRC	Tone Reproduction Curve
UCR	Under Color Removal

### 5.3 Basic numeric types

#### 5.3.1 dateTimeNumber

A 12-byte value representation of the time and date. The actual values are encoded as 16-bit unsigned integers.

**Table 1 — dateTimeNumber**

Byte Offset	Content	Encoded as...
0..1	number of the year (actual year, e.g. 1994)	uInt16Number
2..3	number of the month (1-12)	uInt16Number
4..5	number of the day of the month (1-31)	uInt16Number
6..7	number of hours (0-23)	uInt16Number
8..9	number of minutes (0-59)	uInt16Number
10..11	number of seconds (0-59)	uInt16Number

#### 5.3.2 response16Number

This type is used to associate a normalized device code with a measurement value.

**Table 2 — response16Number**

Byte Offset	Content	Encoded as...
0..1	16-bit number encoding the interval [DeviceMin to DeviceMax] with DeviceMin encoded as 0000h and DeviceMax encoded as FFFFh	uInt16Number
2..3	reserved, must be zero	
4..7	measurement value	s15Fixed16Number



**5.3.3 s15Fixed16Number**

This type represents a fixed signed 4-byte/32-bit quantity which has 16 fractional bits. An example of this encoding is:

**Table 3 — s15Fixed16Number**

-32768.0	80000000h
0	00000000h
1.0	00010000h
$32767 + (65535/65536)$	7FFFFFFFh

**5.3.4 u16Fixed16Number**

This type represents a fixed unsigned 4-byte/32-bit quantity which has 16 fractional bits. An example of this encoding is:

**Table 4 — u16Fixed16Number**

0	00000000h
1.0	00010000h
$65535 + (65535/65536)$	FFFFFFFFh

**5.3.5 u8Fixed8Number**

This type represents a fixed unsigned 2-byte/16-bit quantity which has 8 fractional bits. An example of this encoding is:

**Table 5 — u8Fixed8Number**

0	0000h
1.0	0100h
$255 + (255/256)$	FFFFh

**5.3.6 uint16Number**

This type represents a generic unsigned 2-byte/16-bit quantity.

**5.3.7 uint32Number**

This type represents a generic unsigned 4-byte/32-bit quantity.

**5.3.8 uint64Number**

This type represents a generic unsigned 8-byte/64-bit quantity.

**5.3.9 uint8Number**

This type represents a generic unsigned 1-byte/8-bit quantity.

### 5.3.10 XYZNumber

This type represents a set of three fixed signed 4-byte/32-bit quantities used to encode CIEXYZ tristimulus values where byte usage is assigned as follows:

**Table 6 — XYZNumber**

Byte Offset	Content	Encoded as...
0..3	CIE X	s15Fixed16Number
4..7	CIE Y	s15Fixed16Number
8..11	CIE Z	s15Fixed16Number

For relative tristimulus values, the XYZNumbers are scaled such that a perfect reflecting diffuser has a Y value of 1.0 and not 100.0. Tristimulus values must be non-negative.

### 5.3.11 Seven-bit ASCII

**Table 7 — Hexadecimal**

Hexadecimal															
00	nul	01	soh	02	stx	03	etx	04	eot	05	enq	06	ack	07	bel
08	bs	09	ht	0a	nl	0b	vt	0c	np	0d	cr	0e	so	0f	si
10	dle	11	dc1	12	dc2	13	dc3	14	dc4	15	nak	16	syn	17	etb
18	can	19	em	1a	sub	1b	esc	1c	fs	1d	gs	1e	rs	1f	us
20	sp	21	!	22	"	23	#	24	\$	25	%	26	&	27	'
28	(	29	)	2a	*	2b	+	2c	,	2d	-	2e	.	2f	/
30	0	31	1	32	2	33	3	34	4	35	5	36	6	37	7
38	8	39	9	3a	:	3b	;	3c	<	3d	=	3e	>	3f	?
40	@	41	A	42	B	43	C	44	D	45	E	46	F	47	G
48	H	49	I	4a	J	4b	K	4c	L	4d	M	4e	N	4f	O
50	P	51	Q	52	R	53	S	54	T	55	U	56	V	57	W
58	X	59	Y	5a	Z	5b	[	5c	\	5d	]	5e	^	5f	_
60	`	61	a	62	b	63	c	64	d	65	e	66	f	67	g
68	h	69	i	6a	j	6b	k	6c	l	6d	m	6e	n	6f	o
70	p	71	q	72	r	73	s	74	t	75	u	76	v	77	w
78	x	79	y	7a	z	7b	{	7c		7d	}	7e	~	7f	del

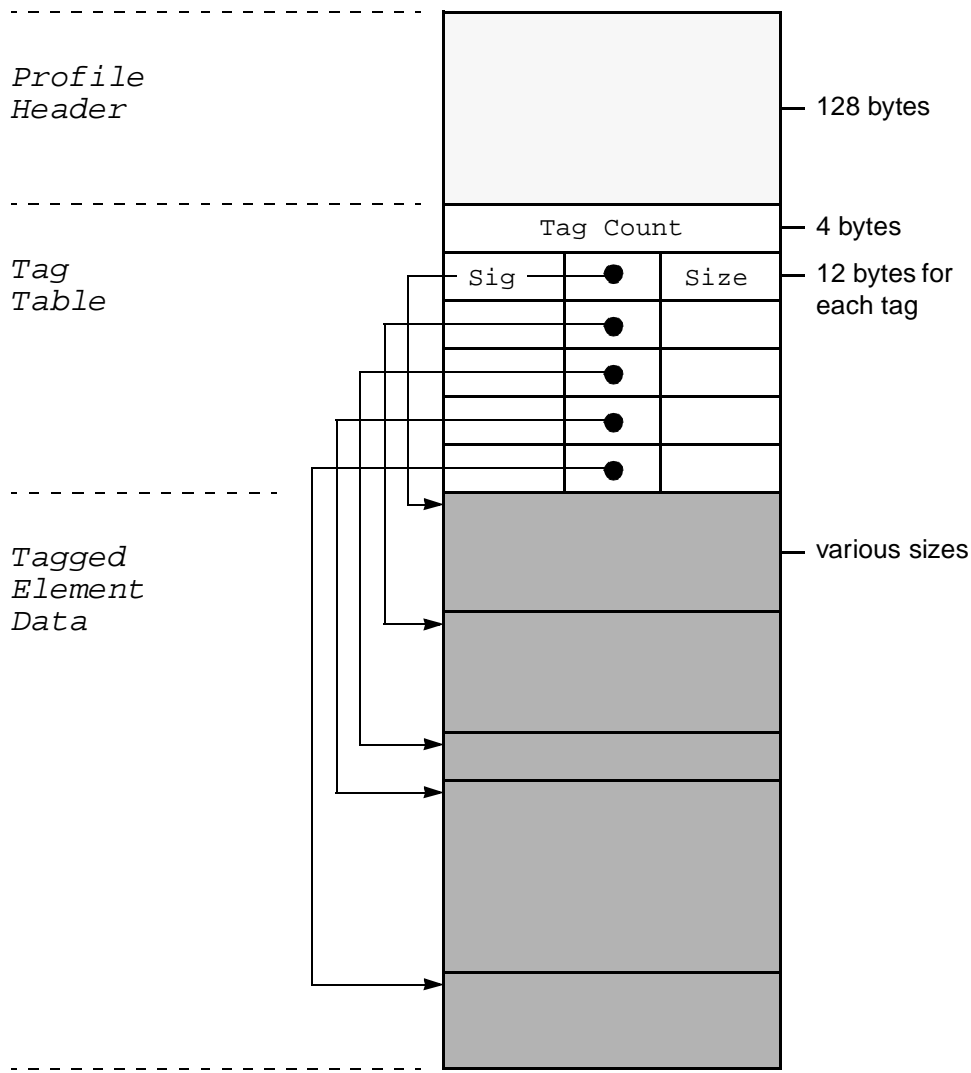
Table 8 — Decimal

Decimal															
0	nul	1	soh	2	stx	3	etx	4	eot	5	enq	6	ack	7	bel
8	bs	9	ht	10	nl	11	vt	12	np	13	cr	14	so	15	si
16	dle	17	dc1	18	dc2	19	dc3	20	dc4	21	nak	22	syn	23	etb
24	can	25	em	26	sub	27	esc	28	fs	29	gs	30	rs	31	us
32	sp	33	!	34	"	35	#	36	\$	37	%	38	&	39	'
40	(	41	)	42	*	43	+	44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[	92	\	93	]	94	^	95	_
96	`	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124		125	}	126	~	127	del

## 6 Requirements

An ICC profile shall include the following elements, in the order shown below in Figure 2, as a single file.

Figure 2 — Profile map



First, the 128-byte file header as defined in 6.1.

Second, the tag table as defined in 6.2.

Third, the tag data elements in accordance with the requirements of 6.3, 6.4 and 6.5.

The information necessary to understand and create the Tag Element Data is arranged within this specification as follows. Each class, and subclass, of device (e.g.: input, RGB) requires the use of specific tags and allows other optional tags. These relationships are described in 6.3. Tags themselves are described in 6.4. However tag descriptions draw upon a series of commonly used "tag types" which are defined in 6.5. The definition of the basic number types used for data encoding are found in 5.3.

All profile data must be encoded as big-endian.

All color spaces used in this specification shall be in accordance with Annex A.

## 6.1 Header description

The profile header provides the necessary information to allow a receiving system to properly search and sort ICC profiles. Table 9 gives the byte position, content and encoding of the profile header.

This header provides a set of parameters at the beginning of the profile format. For color transformation profiles, the device profile dependent fields are set to zero if irrelevant. Having a fixed length header allows for performance enhancements in the profile searching and sorting operations.

**Table 9 — Header**

Byte Offset	Content	Encoded as...
0..3	Profile size	uInt32Number
4..7	CMM Type signature	see below
8..11	Profile version number	see below
12..15	Profile/Device Class signature	see below
16..19	Color space of data (possibly a derived space) [i.e. "the canonical input space"]	see below
20..23	Profile Connection Space (PCS) [i.e. "the canonical output space"]	see below
24..35	Date and time this profile was first created	dateTimeNumber
36..39	'acsp' (61637370h) profile file signature	
40..43	Primary Platform signature	see below
44..47	Flags to indicate various options for the CMM such as distributed processing and caching options	see below
48..51	Device manufacturer of the device for which this profile is created	see below
52..55	Device model of the device for which this profile is created	see below
56..63	Device attributes unique to the particular device setup such as media type	see below
64..67	Rendering Intent	see below
68..79	The XYZ values of the illuminant of the Profile Connection Space. This must correspond to D50. It is explained in more detail in A.1.	XYZNumber
80..83	Profile Creator signature	see below
84..127	44 bytes reserved for future expansion	

### 6.1.1 Profile size

The total size of the profile in bytes.

### 6.1.2 CMM Type signature

Identifies the preferred CMM to be used. The signatures must be registered in order to avoid conflicts (see 0.7). If no CMM is preferred, this field should be set to zero.

### 6.1.3 Profile version

Profile version number where the first 8 bits are the major revision and the next 8 bits are for the minor revision and bug fix revision. The major and minor revision are set by the International Color Consortium and will match up with editions of this specification.

The current profile version number is "2.4.0" (encoded as 02400000h).

The encoding is such that:

**Table 10 — Profile version**

Byte Offset	Content
0	Major Revision in Binary-Coded Decimal
1	Minor Revision & Bug Fix Revision in each nibble in Binary-Coded Decimal
2	reserved, must be set to 0
3	reserved, must be set to 0

A major revision change will only happen when a profile specification change requires that all CMMs to be upgraded in order to correctly use the profile. For example, the addition of new required tags would cause the major revision to change. A minor version change will occur when new profiles can still be used by existing CMMs. For example, the addition of new optional tags would cause the minor revision to change, because existing CMMs will be able to process the profiles correctly while ignoring the new tags.

### 6.1.4 Profile/Device Class signature

There are three basic classes of device profiles: Input, Display and Output profiles.

Within each of these classes there can be a variety of subclasses, such as RGB scanners, CMYK scanners and many others. These basic classes have the following signatures:

**Table 11 — Device class**

Device Class	Signature	Hex Encoding
Input Device profile	'scnr'	73636E72h
Display Device profile	'mnr'	6D6E7472h
Output Device profile	'prtr'	70727472h

In addition to the three basic device profile classes, four additional color processing profiles are defined. These profiles provide a standard implementation for use by the CMM in general color processing or for the convenience of CMMs which may use these types to store calculated transforms. These four profile classes are: device link, color space conversion, abstract, and named color profiles.

DeviceLink profiles provide a mechanism in which to save and store a series of device profiles and non-device profiles in a concatenated format as long as the series begins and ends with a device profile. This

is extremely useful for workflow issues where a combination of device profiles and non-device profiles are used repeatedly.

ColorSpace Conversion profiles are used as a convenient method for CMMs to convert between different non-device color spaces.

The Abstract color profiles provide a generic method for users to make subjective color changes to images or graphic objects by transforming the color data within the PCS.

Named Color profiles can be thought of as sibling profiles to device profiles. For a given device there would be one or more device profiles to handle process color conversions and one or more named color profiles to handle named colors. There might be multiple named color profiles to account for different consumables or multiple named color vendors.

These profiles have the following signatures:

**Table 12 — Profile class**

<b>Profile Class</b>	<b>Signature</b>	<b>Hex Encoding</b>
DeviceLink profile	'link'	6C696E6Bh
ColorSpace Conversion profile	'spac'	73706163h
Abstract profile	'abst'	61627374h
Named Color profile	'nmcl'	6E6D636Ch

### 6.1.5 Color Space signature

The encoding is such that:

**Table 13 — Color space signature (Part 1 of 2)**

<b>Color Space</b>	<b>Signature</b>	<b>Hex Encoding</b>
XYZData	'XYZ '	58595A20h
labData	'Lab '	4C616220h
luvData	'Luv '	4C757620h
YCbCrData	'YCbr '	59436272h
YxyData	'Yxy '	59787920h
rgbData	'RGB '	52474220h
grayData	'GRAY'	47524159h
hsvData	'HSV '	48535620h
hlsData	'HLS '	484C5320h
cmykData	'CMYK'	434D594Bh
cmyData	'CMY '	434D5920h
2colorData	'2CLR'	32434C52h
3colorData (if not listed above)	'3CLR'	33434C52h
4colorData (if not listed above)	'4CLR'	34434C52h

**Table 13 — Color space signature (Part 2 of 2)**

5colorData	'5CLR'	35434C52h
6colorData	'6CLR'	36434C52h
7colorData	'7CLR'	37434C52h
8colorData	'8CLR'	38434C52h
9colorData	'9CLR'	39434C52h
10colorData	'ACLR'	41434C52h
11colorData	'BCLR'	42434C52h
12colorData	'CCLR'	43434C52h
13colorData	'DCLR'	44434C52h
14colorData	'ECLR'	45434C52h
15colorData	'FCLR'	46434C52h

**6.1.6 Profile Connection Space signature**

The encoding is such that:

**Table 14 — Profile connection space signature**

<b>Profile Connection Color Space</b>	<b>Signature</b>	<b>Hex Encoding</b>
XYZData	'XYZ '	58595A20h
labData	'Lab '	4C616220h

When the profile is a DeviceLink profile, the Profile Connection Space Signature is taken from the Color Space Signatures table. (See 6.1.5)

**6.1.7 Primary Platform signature**

Signature to indicate the primary platform/operating system framework for which the profile was created.

The encoding is such that:

**Table 15 — Primary platform signature**

<b>Primary Platform</b>	<b>Signature</b>	<b>Hex Encoding</b>
Apple Computer, Inc.	'APPL'	4150504Ch
Microsoft Corporation	'MSFT'	4D534654h
Silicon Graphics, Inc.	'SGI '	53474920h
Sun Microsystems, Inc.	'SUNW'	53554E57h
Taligent, Inc.	'TGNT'	54474E54h

If there is no primary platform, this field should be set to zero.



### 6.1.8 Profile flags

Flags to indicate various hints for the CMM such as distributed processing and caching options. The least-significant 16 bits are reserved for the ICC.

The encoding is such that:

**Table 16 — Profile flags**

Flags	Bit Position
Embedded Profile (0 if not embedded, 1 if embedded in file)	0
Profile cannot be used independently from the embedded color data (set to 1 if true, 0 if false)	1

### 6.1.9 Device manufacturer and model signatures

The signatures for various manufacturers and models are listed in a separate document (ICC Signatures). New signatures must be registered with the ICC (see 0.7).

### 6.1.10 Attributes

Attributes unique to the particular device setup such as media type. The least-significant 32 bits of this 64-bit value are reserved for the ICC.

The encoding is such that (with “on” having value 1 and “off” having value 0):

**Table 17 — Header attributes**

Attribute	Bit Position
Reflective (off) or Transparency (on)	0
Glossy (off) or Matte (on)	1
Positive (off) or negative (on) media	2
Color (off) or black & white (on) media	3

Notice that bits 0, 1, 2, and 3 describe the media, not the device. For example, a profile for a color scanner that has been loaded with black & white film will have bit 3 set on, regardless of the colorspace of the scanner (see 6.1.5).

If the media is not inherently "color" or "black & white" (such as the paper in an inkjet printer), the reproduction takes on the property of the device. Thus, an inkjet printer loaded with a color ink cartridge can be thought to have "color" media.

### 6.1.11 Rendering intent

Perceptual, media-relative colorimetric, saturation and ICC-absolute colorimetric are the four intents required to be supported. The least-significant 16 bits are reserved for the ICC.

The encoding is such that:

**Table 18 — Header rendering intents**

Rendering Intent	Value
Perceptual	0
Media-Relative Colorimetric	1
Saturation	2
ICC-Absolute Colorimetric	3

The rendering intent specifies the style of reproduction which should be used (or, in the case of a Device-Link profile, was used) when this profile is (was) combined with another profile. In a sequence of profiles, it applies to the combination of this profile and the next profile in the sequence and not to the entire sequence. Typically, the user or application will set the rendering intent dynamically at runtime or embedding time. Therefore, this flag may not have any meaning until the profile is used in some context, e.g in a DeviceLink or an embedded source profile.

#### 6.1.12 Profile Creator signature

Identifies the creator of the profile. The signatures are from the group of signatures used for the device manufacturer field.

## 6.2 Tag table definition

The tag table acts as a table of contents for the tags and tag element data in the profiles. The first four bytes contain a count of the number of tags in the table itself. The tags within the table are not required to be in any particular order.

Each tag signature in the tag table must be unique; a profile cannot contain more than one tag with the same signature.

Individual tag structures within the Tag Table:

**Table 19 — Tag table structure**

Byte Offset	Content	Encoded as...
0..3	Tag Signature	
4..7	Offset to beginning of tag data	uInt32Number
8..11	Element Size	uInt32Number

#### 6.2.1 Tag signature

A four-byte value registered with the ICC (see 0.7).

### 6.2.2 Offset

The address of the tag data element. This is the number of bytes from the beginning of the profile data stream (i.e. the offset to the first byte in the profile header is 0). For profiles that are not embedded in images, this number is the same as the file offset.

All tag data is required to start on a 4-byte boundary (relative to the start of the profile data stream) so that a tag starting with a 32-bit value will be properly aligned without the tag handler needing to know the contents of the tag. This means that the least-significant two bits of each offset must be zero.

### 6.2.3 Element size

The number of bytes in the tag data element. The element size must be for the actual data and must not include any padding at the end of the tag data. An element may have any size (up to the limit imposed by the 32-bit offsets).

## 6.3 Required tags for profiles

This clause provides a top level view of what tags are required for each type of profile classification and a brief description of the algorithmic models associated with these classes. A general description for each tag is included in this clause.

Note that these descriptions assume two things; every profile contains a header, and may include additional tags beyond those listed as required in this clause. The explicitly listed tags are those which are required in order to comprise a legal profile of each type.

In general, multi-dimensional tables refer to lookup tables with more than one input component.

The intent of requiring tags with profiles is to provide a common base level of functionality. If a custom CMM is not present, then the default CMM will have enough information to perform the requested color transformations. The particular models implied by the required data are also described below. While this data might not provide the highest level of quality obtainable with optional data and private data, the data provided is adequate for sophisticated device modeling.

In the following tables, the term "undefined" means that the use of the tag in that situation is not specified by the ICC. The ICC recommends that such tags not be included in profiles. If the tag is present, its use is implementation dependent. In general, the BToAxTags represent the inverse operation of the AToBxTags.

Note that AToB1Tag and BToA1Tag are used to provide both of the colorimetric intents.

Note that tags may reference the same tag data.

If an optional transformation tag is not present, the general fall back strategy of the CMM is implementation dependent.

The interpretation of some tags are context dependent. This dependency is described below in Table 20.

**Table 20 — Profile type/profile tag and defined rendering intents**

Profile Class	AToB0Tag	AToB1Tag	AToB2Tag	TRC/ matrix	BToA0Tag	BToA1Tag	BToA2Tag
<b>Input</b>	Device to PCS: perceptual	Device to PCS: colorimetric	Device to PCS: saturation	colorimetric	PCS to Device: perceptual	PCS to Device: colorimetric	PCS to Device: saturation
<b>Display</b>	Device to PCS: perceptual	Device to PCS: colorimetric	Device to PCS: saturation	colorimetric	PCS to Device: perceptual	PCS to Device: colorimetric	PCS to Device: saturation
<b>Output</b>	Device to PCS: perceptual	Device to PCS: colorimetric	Device to PCS: saturation	undefined	PCS to Device: perceptual	PCS to Device: colorimetric	PCS to Device: saturation
<b>Color-Space</b>	ColorSpace to PCS: perceptual	ColorSpace to PCS: colorimetric	ColorSpace to PCS: saturation	undefined	PCS to Color-Space: perceptual	PCS to Color-Space: colorimetric	PCS to Color-Space: saturation
<b>Abstract</b>	PCS to PCS	undefined	undefined	undefined	undefined	undefined	undefined
<b>DeviceLink</b>	Device1 to Device2 rendering intent defined according to Table 9	undefined	undefined	undefined	undefined	undefined	undefined
<b>Named Color</b>	undefined	undefined	undefined	undefined	undefined	undefined	undefined

### 6.3.1 Input Profile

This profile represents input devices such as scanners and digital cameras.

#### 6.3.1.1 Monochrome Input Profiles

**Table 21 — Monochrome input profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
grayTRCTag	Gray tone reproduction curve (TRC)
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

The mathematical model implied by this data is:

$$connection = grayTRC[device] \quad (1)$$

This represents a simple tone reproduction curve adequate for most monochrome input devices. The *connection* values in this equation should represent the achromatic channel of the profile connection space. If the inverse of this is desired, then the following equation is used:

$$device = grayTRC^{-1}[connection] \quad (2)$$

AToB0Tag, AToB1Tag, AToB2Tag, BToA0Tag, BToA1Tag, BToA2Tag may be included in monochrome profiles. If these are present, their usage shall be as defined in Table 20.

### 6.3.1.2 Three-component Matrix-based Input Profiles

This profile type is often used with devices whose color space is RGB. However, any three-component color space can be used.

**Table 22 — Three-component matrix-based input profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
redColorantTag	Red colorant XYZ relative tristimulus values
greenColorantTag	Green colorant XYZ relative tristimulus values
blueColorantTag	Blue colorant XYZ relative tristimulus values
redTRCTag	Red channel tone reproduction curve
greenTRCTag	Green channel tone reproduction curve
blueTRCTag	Blue channel tone reproduction curve
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

The forward mathematical model implied by this data is:

$$\begin{aligned}
 linear_r &= redTRC[device_r] \\
 linear_g &= greenTRC[device_g] \\
 linear_b &= blueTRC[device_b]
 \end{aligned}
 \tag{3}$$

$$\begin{bmatrix} connection_X \\ connection_Y \\ connection_Z \end{bmatrix} = \begin{bmatrix} redColorant_X & greenColorant_X & blueColorant_X \\ redColorant_Y & greenColorant_Y & blueColorant_Y \\ redColorant_Z & greenColorant_Z & blueColorant_Z \end{bmatrix} \begin{bmatrix} linear_r \\ linear_g \\ linear_b \end{bmatrix}
 \tag{4}$$

This represents a simple linearization followed by a linear mixing model. The three tone reproduction curves linearize the raw values with respect to the luminance (Y) dimension of the CIEXYZ encoding of the profile connection space. The 3x3 matrix converts these linearized values into XYZ values for the CIEXYZ encoding of the profile connection space. The inverse model is given by the following equations:

$$\begin{bmatrix} linear_r \\ linear_g \\ linear_b \end{bmatrix} = \begin{bmatrix} redColorant_X & greenColorant_X & blueColorant_X \\ redColorant_Y & greenColorant_Y & blueColorant_Y \\ redColorant_Z & greenColorant_Z & blueColorant_Z \end{bmatrix}^{-1} \begin{bmatrix} connection_X \\ connection_Y \\ connection_Z \end{bmatrix} \quad (5)$$

$$\begin{aligned} device_r &= redTRC^{-1}[linear_r] \\ device_g &= greenTRC^{-1}[linear_g] \\ device_b &= blueTRC^{-1}[linear_b] \end{aligned} \quad (6)$$

Only the CIEXYZ encoding of the profile connection space can be used with matrix/TRC models. This profile may be used for any device which has a three-component color space suitably related to XYZ by this model. If the color space is not RGB, then  $device_r$ ,  $device_g$ , and  $device_b$  in equations (3) and (6) are replaced by the proper device components.

An AToB0Tag must be included if the CIELAB encoding of the profile connection space is to be used.

Additional multidimensional tags (AToB0Tag, AToB1Tag, AToB2Tag, BToA0Tag, BToA1Tag, BToA2Tag) may also be included. If these are present, their usage shall be as defined in Table 20.

In addition, a gamutTag may be included. The usage of this tag is identical as in output profiles (Section 6.3.3.2).

### 6.3.1.3 N-component LUT-based Input Profiles

**Table 23 — N-component LUT-based input profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
AToB0Tag	Device to PCS: 8-bit or 16-bit data
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

The AToB0Tag represents a device model described by the lut8Type or lut16Types. This tag provides the parameter data for an algorithm that includes a set of non-interdependent per-channel tone reproduction curves, a multi-dimensional lookup table and a set of non-interdependent per-channel linearization curves. The mathematical model implied by this data is described in detail in 6.5.7 and 6.5.8 that specify the general lookup table tag element structures.

Additional multidimensional tags (AToB1Tag, AToB2Tag, BToA0Tag, BToA1Tag, BToA2Tag) may also be included. If these are present, their usage shall be as defined in Table 20.

In addition, a gamutTag may be included. The usage of this tag is identical as in output profiles (Section 6.3.3.2).

### 6.3.2 Display Profile

This profile represents display devices such as monitors.

#### 6.3.2.1 Monochrome Display Profiles

**Table 24 — Monochrome display profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
grayTRCTag	Gray tone reproduction curve
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

The mathematical model implied by this data is:

$$connection = grayTRC[device] \quad (7)$$

This represents a simple tone reproduction curve adequate for most monochrome display devices. The *connection* values in this equation should represent the achromatic channel of the profile connection space. If the inverse of this is desired, then the following equation is used:

$$device = grayTRC^{-1}[connection] \quad (8)$$

AToB0Tag, AToB1Tag, AToB2Tag, BToA0Tag, BToA1Tag, BToA2Tag may be included in monochrome profiles. If these are present, their usage shall be as defined in Table 20.

## 6.3.2.2 RGB Display Profiles

Table 25 — RGB display profile required tags

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
redColorantTag	Relative XYZ values of red phosphor
greenColorantTag	Relative XYZ values of green phosphor
blueColorantTag	Relative XYZ values of blue phosphor
redTRCTag	Red channel tone reproduction curve
greenTRCTag	Green channel tone reproduction curve
blueTRCTag	Blue channel tone reproduction curve
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

This model is based on a three non-interdependent per-channel tone reproduction curves to convert between linear and non-linear RGB values and a 3x3 matrix to convert between linear RGB values and relative XYZ values. The mathematical model implied by this data is:

$$\begin{aligned}
 linear_r &= redTRC[device_r] \\
 linear_g &= greenTRC[device_g] \\
 linear_b &= blueTRC[device_b]
 \end{aligned}
 \tag{9}$$

$$\begin{bmatrix} connection_X \\ connection_Y \\ connection_Z \end{bmatrix} = \begin{bmatrix} redColorant_X & greenColorant_X & blueColorant_X \\ redColorant_Y & greenColorant_Y & blueColorant_Y \\ redColorant_Z & greenColorant_Z & blueColorant_Z \end{bmatrix} \begin{bmatrix} linear_r \\ linear_g \\ linear_b \end{bmatrix}
 \tag{10}$$

This represents a simple linearization followed by a linear mixing model. The three tone reproduction curves linearize the raw values with respect to the luminance (Y) dimension of the CIEXYZ encoding of the profile connection space. The 3x3 matrix converts these linearized values into XYZ values for the CIEXYZ encoding of the profile connection space. The inverse model is given by the following equations:

$$\begin{bmatrix} linear_r \\ linear_g \\ linear_b \end{bmatrix} = \begin{bmatrix} redColorant_X & greenColorant_X & blueColorant_X \\ redColorant_Y & greenColorant_Y & blueColorant_Y \\ redColorant_Z & greenColorant_Z & blueColorant_Z \end{bmatrix}^{-1} \begin{bmatrix} connection_X \\ connection_Y \\ connection_Z \end{bmatrix}
 \tag{11}$$



$$\begin{aligned}
 device_r &= redTRC^{-1}[linear_r] \\
 device_g &= greenTRC^{-1}[linear_g] \\
 device_b &= blueTRC^{-1}[linear_b]
 \end{aligned}
 \tag{12}$$

Only the CIEXYZ encoding of the profile connection space can be used with matrix/TRC models. A multidimensional table tag must be included if the CIELAB encoding of the profile connection space is to be used.

Additional multidimensional tags (AToB0Tag, AToB1Tag, AToB2Tag, BToA0Tag, BToA1Tag, BToA2Tag) may also be included. If these are present, their usage shall be as defined in Table 20.

In addition, a gamutTag may be included. The usage of this tag is identical as in output profiles (Section 6.3.3.2).

### 6.3.3 Output Profile

This profile represents output devices such as printers and film recorders. The LUT tags that are required by the printer profiles contain either the 8-bit or the 16-bit LUTs as described in 6.5.7 and 6.5.8. The LUT algorithm for profile connection space to device space transformations process data sequentially through a matrix, input tables, a color LUT, and output tables.

#### 6.3.3.1 Monochrome Output Profiles

**Table 26 — Monochrome output profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
grayTRCTag	Gray tone reproduction curve
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

The mathematical model implied by this data is:

$$connection = grayTRC[device] \tag{13}$$

This represents a simple tone reproduction curve adequate for most monochrome output devices. The *connection* values in this equation should represent the achromatic channel of the profile connection space. If the inverse of this is desired, then the following equation is used:

$$device = grayTRC^{-1}[connection] \tag{14}$$

AToB0Tag, AToB1Tag, AToB2Tag, BToA0Tag, BToA1Tag, BToA2Tag may be included in monochrome profiles. If these are present, their usage shall be as defined in Table 20.

Note: The output values are the control values and not the "K" (black) values.

### 6.3.3.2 Color Output Profiles

**Table 27 — Color output profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
AToB0Tag	Device to PCS: 8-bit or 16-bit data: intent of 0
BToA0Tag	PCS to Device space: 8-bit or 16-bit data: intent of 0
gamutTag	Out of Gamut: 8-bit or 16-bit data
AToB1Tag	Device to PCS: 8-bit or 16-bit data: intent of 1
BToA1Tag	PCS to Device space: 8-bit or 16-bit data: intent of 1
AToB2Tag	Device to PCS: 8-bit or 16 bit-data: intent of 2
BToA2Tag	PCS to Device space: 8-bit or 16-bit data: intent of 2
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

These tags represent a device model described in 6.5.8 or 6.5.7. The intent values described in these tags directly correlate to the value of the rendering intent header flag of the source profile in the color modeling session (See Table 18).

Each of the first three intents are associated with a specific tag. The fourth intent, ICC-absolute colorimetry, is obtained by modifying the media-relative colorimetric intent tag based on the values which are in the mediaWhitePointTag. It is permissible to reference the same tag for all of these intents and to use the media-relative colorimetric intent tag when ICC-absolute colorimetry is specified.

In essence, each of these tags provides the parameter data for an algorithm that includes a 3x3 matrix, a set of non-interdependent per-channel tone reproduction curves, a multidimensional lookup table and a set of non-interdependent per-channel linearization curves. The algorithmic details of this model and the intent of each tag is given later in 6.5.7 or 6.5.8 that specify the general lookup table tag element structures.

### 6.3.4 Additional Profile Formats

#### 6.3.4.1 Device Link Profile

This profile represents a one-way link or connection between devices. It does not represent any device model nor can it be embedded into images.

**Table 28 — Device link profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
AToB0Tag	Actual transformation parameter structure; 8-bit or 16-bit data
profileSequenceDescTag	An array of descriptions of the profile sequence
copyrightTag	7-bit ASCII profile copyright information

The single AToB0Tag may contain any of the four possible rendering intents. The rendering intent used is indicated in the header of the profile.

The AToB0Tag represents a device model described in 6.5.7 or 6.5.8. This tag provides the parameter data for an algorithm that includes a 3x3 matrix, a set of non-interdependent per-channel tone reproduction curves, a multidimensional lookup table and a set of non-interdependent per-channel linearization curves. The algorithmic details of this model and the intent of each tag is given later in 6.5.7 and 6.5.8 that specify the general lookup table tag element structures. This is a pre-evaluated transform that cannot be undone.

The color space of data in the DeviceLink profile will be the same as the color space of the data of the first profile in the sequence. The profile connection space will be the same as the color space of the data of the last profile in the sequence.

#### 6.3.4.2 ColorSpace Conversion Profile

This profile provides the relevant information to perform a color space transformation between the non-device color spaces and the PCS. It does not represent any device model. ColorSpace Conversion profiles may be embedded in images.

**Table 29 — ColorSpace conversion profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
BToA0Tag	Inverse transformation parameter structure; 8-bit or 16-bit data
AToB0Tag	Actual transformation parameter structure; 8-bit or 16-bit data
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

The AToB0Tag and BToA0Tag represent a model described in 6.5.7 or 6.5.8. This tag provides the parameter data for an algorithm that includes a 3x3 matrix, a set of non-interdependent per-channel tone reproduction curves, a multidimensional lookup table and a set of non-interdependent per-channel linearization curves. The algorithmic details of this model and the intent of each tag is given later in 6.5.7 and 6.5.8 that specify the general lookup table tag element structures.

For color transformation profiles, the device profile dependent fields are set to zero if irrelevant.

Additional multidimensional tags (AToB1Tag, AToB2Tag., BToA1Tag, BToA2Tag) may also be included. If these are present, their usage shall be as defined in Table 20.

In addition, a gamutTag may be included. The usage of this tag is identical as in output profiles (Section 6.3.3.2).

### 6.3.4.3 Abstract Profile

This profile represents abstract transforms and does not represent any device model. Color transformations using abstract profiles are performed from PCS to PCS. Abstract profiles cannot be embedded in images.

**Table 30 — Abstract profile required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
AToB0Tag	Actual transformation parameter structure; 8-bit or 16-bit data
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

The AToB0Tag represents a PCS to PCS model described by the lut8Type or lut16Types. This tag provides the parameter data for an algorithm that includes a 3x3 matrix, a set of non-interdependent per-channel tone reproduction curves, a multidimensional lookup table and a set of non-interdependent per-channel linearization curves. The algorithmic details of this model and the intent of each tag is given later in 6.5.7 and 6.5.8 that specify the general lookup table tag element structures.

### 6.3.4.4 Named Color Profile

Named color profiles can be thought of as sibling profiles to device profiles. For a given device there would be one or more device profiles to handle process color conversions and one or more named color profiles to handle named colors. There might be multiple named color profiles to account for different consumables or multiple named color vendors. This profile provides a PCS and optional device representation for a list of named colors. Named color profiles are device-specific in that their data is shaped for a particular device.

**Table 31 — Named color required tags**

Tag Name	General Description
profileDescriptionTag	Structure containing invariant and localizable versions of the profile name for display
namedColor2Tag	PCS and optional device representation for named colors
mediaWhitePointTag	Media XYZ white point
copyrightTag	7-bit ASCII profile copyright information

The namedColor2Tag provides a PCS and optional device representation for each named color in a list of named colors. The PCS representation is provided to support general color management functionality. It is very useful for display and emulation of the named colors.

When using a named color profile with the device for which it is intended, the device representation of the color specifies the exact device coordinates for each named color. The PCS representation in conjunction with the device's output profile can provide an approximation of these exact coordinates. The exactness of this approximation is a function of the accuracy of the output profile and the color management system performing the transformations.

The combination of the PCS and device representations provides for flexibility with respect to accuracy and portability.

## 6.4 Tag descriptions

This clause specifies the individual tags used to create all possible portable profiles in the ICC Profile Format. The appropriate tag typing is indicated with each individual tag description. Note that the signature indicates only the type of data and does not imply anything about the use or purpose for which the data is intended.

In addition to the tags listed below, any of the previously defined tags in 6.3 can also be used as optional tags if they are not used in the required set for a particular profile and are not specifically excluded in a profile definition.

**Table 32 — Tag list (Part 1 of 2)**

<b>Tag Name</b>	<b>General Description</b>
AToB0Tag	Multidimensional transformation structure
AToB1Tag	Multidimensional transformation structure
AToB2Tag	Multidimensional transformation structure
blueColorantTag	Relative XYZ values of blue phosphor or colorant
blueTRCTag	Blue channel tone reproduction curve
BToA0Tag	Multidimensional transformation structure
BToA1Tag	Multidimensional transformation structure
BToA2Tag	Multidimensional transformation structure
calibrationDateTimeTag	Profile calibration date and time
charTargetTag	Characterization target such as IT8/7.2
chromaticAdaptationTag	Multidimensional transform for non-D50 adaptation
chromaticityTag	Set of phosphor/colorant chromaticity
copyrightTag	7-bit ASCII profile copyright information
crdInfoTag	Names of companion CRDs to the profile
deviceMfgDescTag	Displayable description of device manufacturer
deviceModelDescTag	Displayable description of device model
deviceSettingsTag	Specifies the device settings for which the profile is valid
gamutTag	Out of gamut: 8-bit or 16-bit data
grayTRCTag	Gray tone reproduction curve
greenColorantTag	Relative XYZ values of green phosphor or colorant
greenTRCTag	Green channel tone reproduction curve
luminanceTag	Absolute luminance for emissive device
measurementTag	Alternative measurement specification information
mediaBlackPointTag	Media XYZ black point
mediaWhitePointTag	Media XYZ white point

**Table 32 — Tag list (Part 2 of 2)**

outputResponseTag	Description of the desired device response
preview0Tag	Preview transformation: 8-bit or 16-bit data
preview1Tag	Preview transformation: 8-bit or 16-bit data
preview2Tag	Preview transformation: 8-bit or 16-bit data
profileDescriptionTag	Profile description for display
profileSequenceDescTag	Profile sequence description from source to destination
ps2CRD0Tag	PostScript Level 2 color rendering dictionary: perceptual
ps2CRD1Tag	PostScript Level 2 color rendering dictionary: colorimetric
ps2CRD2Tag	PostScript Level 2 color rendering dictionary: saturation
ps2CRD3Tag	PostScript Level 2 color rendering dictionary: ICC-absolute
ps2CSATag	PostScript Level 2 color space array
ps2RenderingIntentTag	PostScript Level 2 Rendering Intent
redColorantTag	Relative XYZ values of red phosphor or colorant
redTRCTag	Red channel tone reproduction curve
screeningDescTag	Screening attributes description
screeningTag	Screening attributes such as frequency, angle and spot shape
technologyTag	Device technology information such as LCD, CRT, Dye Sublimation, etc.
ucrbgTag	Under color removal and black generation description
viewingCondDescTag	Viewing condition description
viewingConditionsTag	Viewing condition parameters

**6.4.1 AToB0Tag**

Tag Type: lut8Type or lut16Type  
 Tag Signature: 'A2B0' (41324230h)

Device to PCS: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

**6.4.2 AToB1Tag**

Tag Type: lut8Type or lut16Type  
 Tag Signature: 'A2B1' (41324231h)

Device to PCS: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

**6.4.3 AToB2Tag**

Tag Type: lut8Type or lut16Type  
 Tag Signature: 'A2B2' (41324232h)

Device to PCS: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

#### **6.4.4 blueColorantTag**

Tag Type: XYZType  
Tag Signature: 'bXYZ' (6258595Ah)

The relative XYZ values of blue phosphor or colorant.

#### **6.4.5 blueTRCTag**

Tag Type: curveType  
Tag Signature: 'bTRC' (62545243h)

Blue channel tone reproduction curve. The first element represents no colorant (white) or phosphors (black) and the last element represents 100 percent colorant (blue) or 100 percent phosphor (blue).

#### **6.4.6 BToA0Tag**

Tag Type: lut8Type or lut16Type  
Tag Signature: 'B2A0' (42324130h)

PCS to Device space: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

#### **6.4.7 BToA1Tag**

Tag Type: lut8Type or lut16Type  
Tag Signature: 'B2A1' (42324131h)

PCS to Device space: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

#### **6.4.8 BToA2Tag**

Tag Type: lut8Type or lut16Type  
Tag Signature: 'B2A2' (42324132h)

PCS to Device space: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

#### **6.4.9 calibrationDateTimeTag**

Tag Type: dateTimeType  
Tag Signature: 'calt' (63616C74h)

Profile calibration date and time. Initially, this tag matches the contents of the profile header's creation date/time field. This allows applications and utilities to verify if this profile matches a vendor's profile and how recently calibration has been performed.

#### **6.4.10 charTargetTag**

Tag Type: textType  
Tag Signature: 'targ' (74617267h)

This tag contains the measurement data for a characterization target such as IT8.7/2. This tag is provided so that distributed utilities can create transforms "on the fly" or check the current performance against the

original device performance. The tag embeds the exact data file format defined in the ANSI or ISO standard which is applicable to the device being characterized.

Examples are the data formats described in ANSI IT8.7/1-1993 section 4.10 and ANSI IT8.7/2-1993 section 4.10. Each of these file formats contains an identifying character string as the first few bytes of the format, allowing an external parser to determine which data file format is being used. This provides the facilities to include a wide range of targets using a variety of measurement specifications in a standard manner.

Note: The IT8 specifications do not currently have a keyword which identifies the set as being reference data as opposed to device response data. An addition to enable this additional data set is being considered by the IT8 committee.

#### 6.4.11 chromaticAdaptationTag

Tag Type: s15Fixed16ArrayType  
Tag Signature: 'chad' (63686164h)

This tag converts the input XYZ color, measured at a device's specific viewing conditions, to the output XYZ color in the PCS viewing conditions after complete adaptation.

The tag reflects the survey of the currently used methods of conversion, all of which can be formulated as a matrix transformation. Such a 3 by 3 chromatic adaptation matrix is organized as a 9-element array of signed 15.16 numbers (s15Fixed16ArrayType tag). Similarly as in the other occurrences of a 3 by 3 matrix in the ICC tags, the dimension corresponding to the matrix rows varies least rapidly while the one corresponding to the matrix columns varies most rapidly.

$$array = [a0 \ a1 \ a2 \ a3 \ a4 \ a5 \ a6 \ a7 \ a8] \quad (15)$$

$$\begin{bmatrix} X_{pcs} \\ Y_{pcs} \\ Z_{pcs} \end{bmatrix} = \begin{bmatrix} a0 & a1 & a2 \\ a3 & a4 & a5 \\ a6 & a7 & a8 \end{bmatrix} \begin{bmatrix} X_{src} \\ Y_{src} \\ Z_{src} \end{bmatrix} \quad (16)$$

Where  $XYZ_{src}$  represents the value in the actual device viewing condition and  $XYZ_{pcs}$  represents the chromatically adapted value in the reference viewing condition.

The chromatic adaptation matrix is a combination of three separate conversions:

- 1) Conversion of source CIE XYZ tristimulus values to cone response tristimulus values.
- 2) Adjustment of the cone response values for an observer's chromatic adaptation.
- 3) Conversion of the adjusted cone response tristimulus back to CIE XYZ values.

#### 6.4.12 chromaticityTag

Tag Type: chromaticityType  
Tag Signature: 'chrm' (6368726Dh)



The data and type of phosphor/colorant chromaticity set.

#### **6.4.13 copyrightTag**

Tag Type: textType  
Tag Signature: 'cppt' (63707274h)

This tag contains the 7-bit ASCII text copyright information for the profile.

#### **6.4.14 crdInfoTag**

Tag Type: crdInfoType  
Tag Signature: 'crdi' (63726469h)

This tag contains the PostScript product name to which this profile corresponds and the names of the companion CRDs. Recall that a single profile can generate multiple CRDs.

See C.1 for information about using this tag.

#### **6.4.15 deviceMfgDescTag**

Tag Type: textDescriptionType  
Tag Signature: 'dmnd' (646D6E64h)

Structure containing invariant and localizable versions of the device manufacturer for display. The content of this structure is described in 6.5.17.

#### **6.4.16 deviceModelDescTag**

Tag Type: textDescriptionType  
Tag Signature: 'dmdd' (646D6464h)

Structure containing invariant and localizable versions of the device model for display. The content of this structure is described in 6.5.17.

#### **6.4.17 deviceSettingsTag**

Tag Type: deviceSettingsType  
Tag Signature: 'devs' (64657673h)

This tag specifies the device settings for which the profile is valid.

These settings are usually specific to a platform, so they are specified for each platform separately. The profile creator decides which platforms need to be supported.

The profile may be valid only for certain combination of settings like a specific media type with a specific resolution with a specific halftone. Therefore, for each platform there are groupings or combinations of settings.

Under each grouping, multiple values can be specified for each setting. For example, if the profile is valid for resolutions *a* and *b* with media type *c* and halftone style *d*, then "resolution" will have two values, while the other two settings have one value each in the combination.

The content of this structure is described in 6.5.6.

#### **6.4.18 gamutTag**

Tag Type: lut8Type or lut16Type  
Tag Signature: 'gamt' (67616D74h)

Out of gamut tag: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

This tag takes PCS values as its input and produces a single channel of output. If the output value is 0, the PCS color is in-gamut. If the output is non-zero, the PCS color is out-of-gamut, with the number "n+1" being at least as far out of gamut as the number "n".

#### **6.4.19 grayTRCTag**

Tag Type: curveType  
Tag Signature: 'kTRC' (6B545243h)

Gray tone reproduction curve. The tone reproduction curve provides the necessary information to convert between a single device channel and the CIEXYZ encoding of the profile connection space. The first element represents no colorant (white) or phosphors (black) and the last element represents 100 percent colorant (black) or 100 percent phosphor (white).

#### **6.4.20 greenColorantTag**

Tag Type: XYZType  
Tag Signature: 'gXYZ' (6758595Ah)

Relative XYZ values of green phosphor or colorant.

#### **6.4.21 greenTRCTag**

Tag Type: curveType  
Tag Signature: 'gTRC' (67545243h)

Green channel tone reproduction curve. The first element represents no colorant (white) or phosphors (black) and the last element represents 100 percent colorant (green) or 100 percent phosphor (green).

#### **6.4.22 luminanceTag**

Tag Type: XYZType  
Tag Signature: 'lumi' (6C756D69h)

Absolute luminance of emissive devices in candelas per square meter as described by the Y channel. The X and Z channels are ignored in all cases.

#### **6.4.23 measurementTag**

Tag Type: measurementType  
Tag Signature: 'meas' (6D656173h)

Alternative measurement specification such as a D65 illuminant instead of the default D50.

#### 6.4.24 mediaBlackPointTag

Tag Type: XYZType  
Tag Signature: 'bkpt' (626B7074h)

This tag specifies the media black point. It is referenced to the profile connection space so that the media black point as represented in the PCS is equivalent to this tag value. If this tag is not present, it is assumed to be (0,0,0).

#### 6.4.25 mediaWhitePointTag

Tag Type: XYZType  
Tag Signature: 'wtpt' (77747074h)

This tag specifies the media white point and is used for generating ICC-absolute colorimetric intent. See Annex A for a more complete description of its use.

#### 6.4.26 namedColorTag

Tag Type: namedColorType  
Tag Signature: 'ncol' (6E636F6Ch)

NOTE: This tag is obsolete, and should not be used in new profiles. Use namedColor2Tag instead.

Named color reference transformation for converting between named color sets and the profile connection space or device color spaces.

#### 6.4.27 namedColor2Tag

Tag Type: namedColor2Type  
Tag Signature: 'ncl2' (6E636C32h)

Named color information providing a PCS and optional device representation for a list of named colors. The namedColorTag should no longer be used.

#### 6.4.28 outputResponseTag

Tag Type: responseCurveSet16Type  
Tag Signature: 'resp' (72657370h)

Structure containing a description of the device response for which the profile is intended. The content of this structure is described in 6.5.13.

**NOTE:** The user's attention is called to the possibility that the use of this tag for device calibration may require use of an invention covered by patent rights. By publication of this specification, no position is taken with respect to the validity of this claim or of any patent rights in connection therewith. The patent holder has, however, filed a statement of willingness to grant a license under these rights on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license. Details may be obtained from the publisher.

#### 6.4.29 preview0Tag

Tag Type: lut8Type or lut16Type  
Tag Signature: 'pre0' (70726530h)

Preview transformation from PCS to device space and back to the PCS: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

#### **6.4.30 preview1Tag**

Tag Type: lut8Type or lut16Type  
Tag Signature: 'pre1' (70726531h)

Preview transformation from the PCS to device space and back to the PCS: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

#### **6.4.31 preview2Tag**

Tag Type: lut8Type or lut16Type  
Tag Signature: 'pre2' (70726532h)

Preview transformation from PCS to device space and back to the PCS: 8-bit or 16-bit data. The processing mechanisms are described in 6.5.7 and 6.5.8.

#### **6.4.32 profileDescriptionTag**

Tag Type: textDescriptionType  
Tag Signature: 'desc' (64657363h)

Structure containing invariant and localizable versions of the profile description for display. The content of this structure is described in 6.5.17. This invariant description has no fixed relationship to the actual profile disk file name.

#### **6.4.33 profileSequenceDescTag**

Tag Type: profileSequenceDescType  
Tag Signature: 'pseq' (70736571h)

Structure containing a description of the profile sequence from source to destination, typically used with the DeviceLink profile. The content of this structure is described in 6.5.12.

#### **6.4.34 ps2CRD0Tag**

Tag Type: dataType  
Tag Signature: 'psd0' (70736430h)

PostScript Level 2 Type 1 color rendering dictionary (CRD) for the Perceptual rendering intent. This tag provides the dictionary operand to the setcolorrendering operator. This tag can be used in conjunction with the setcolorrendering operator on any PostScript Level 2 device.

See Annex C for the relationship between the ICC profile data and PostScript Tags.

#### **6.4.35 ps2CRD1Tag**

Tag Type: dataType  
Tag Signature: 'psd1' (70736431h)

PostScript Level 2 Type 1 CRD for the media-relative Colorimetric rendering intent. This tag provides the dictionary operand to the setcolorrendering operator. This tag can be used in conjunction with the setcolorrendering operator on any PostScript Level 2 device.

See Annex C for the relationship between the ICC profile data and PostScript Tags.

#### **6.4.36 ps2CRD2Tag**

Tag Type: dataType

Tag Signature: 'psd2' (70736432h)

PostScript Level 2 Type 1 CRD for the Saturation rendering intent. This tag provides the dictionary operand to the setcolorrendering operator. This tag can be used in conjunction with the setcolorrendering operator on any PostScript Level 2 device.

See Annex C for the relationship between the ICC profile data and PostScript Tags.

#### **6.4.37 ps2CRD3Tag**

Tag Type: dataType

Tag Signature: 'psd3' (70736433h)

PostScript Level 2 Type 1 CRD for the ICC-Absolute Colorimetric rendering intent. This tag provides the dictionary operand to the setcolorrendering operator. This tag can be used in conjunction with the setcolorrendering operator on any PostScript Level 2 device.

See Annex C for the relationship between the ICC profile data and PostScript Tags.

#### **6.4.38 ps2CSATag**

Tag Type: dataType

Tag Signature: 'ps2s' (70733273h)

PostScript Level 2 color space array. This tag provides the array operand to the setcolorspace operator. For color spaces that fit within the original PostScript Level 2 device independent color model no operator verification need be performed. For color spaces that fit only within extensions to this model, operator verification is first required. An example of this would be for Calibrated CMYK input color spaces which are supported via an extension. In such cases where the necessary PostScript Level 2 support is not available, PostScript Level 1 color spaces, such as DeviceCMYK, can be used, or the colors can be converted on the host using a CMS. In the latter case, the PostScript Level 1 color operators are used to specify the device dependent (pre-converted) colors. The PostScript contained in this tag expects the associated color values instantiated either through setcolor or image to be in the range [0, 1].

See Annex C for the relationship between the ICC profile data and PostScript Tags.

#### **6.4.39 ps2RenderingIntentTag**

Tag Type: dataType

Tag Signature: 'ps2i' (70733269h)

PostScript Level 2 rendering intent. This tag provides the operand to the findcolorrendering operator. findcolorrendering is not necessarily supported on all PostScript Level 2 devices, hence its existence must first be established. Standard values for ps2RenderingIntentTag are media-relative colorimetric, ICC-abso-

lute colorimetric, perceptual, and saturation. These intents are meant to correspond to the rendering intents of the profile's header.

See Annex C for the relationship between the ICC profile data and PostScript Tags.

#### 6.4.40 redColorantTag

Tag Type: XYZType

Tag Signature: 'rXYZ' (7258595Ah)

Relative XYZ values of red phosphor or colorant.

#### 6.4.41 redTRCTag

Tag Type: curveType

Tag Signature: 'rTRC' (72545243h)

Red channel tone reproduction curve. The first element represents no colorant (white) or phosphors (black) and the last element represents 100 percent colorant (red) or 100 percent phosphor (red).

#### 6.4.42 screeningDescTag

Tag Type: textDescriptionType

Tag Signature: 'scrd' (73637264h)

Structure containing invariant and localizable versions of the screening conditions. The content of this structure is described in 6.5.17.

#### 6.4.43 screeningTag

Tag Type: screeningType

Tag Signature: 'scrn' (7363726Eh)

This tag contains screening information for a variable number of channels.

#### 6.4.44 technologyTag

Tag Type: signatureType

Tag Signature: 'tech' (74656368h)

Device technology information such as CRT, Dye Sublimation, etc. The encoding is such that:

**Table 33 — Technology signatures**

Technology	Signature	Hex Encoding
Film Scanner	'fscn'	6673636Eh
Digital Camera	'dcam'	6463616Dh
Reflective Scanner	'rscn'	7273636Eh
Ink Jet Printer	'ijet'	696A6574h
Thermal Wax Printer	'twax'	74776178h
Electrophotographic Printer	'epho'	6570686Fh

**Table 33 — Technology signatures**

Electrostatic Printer	'esta'	65737461h
Dye Sublimation Printer	'dsub'	64737562h
Photographic Paper Printer	'rpho'	7270686Fh
Film Writer	'fprn'	6670726Eh
Video Monitor	'vidm'	7669646Dh
Video Camera	'vidc'	76696463h
Projection Television	'pjtv'	706A7476h
Cathode Ray Tube Display	'CRT '	43525420h
Passive Matrix Display	'PMD '	504D4420h
Active Matrix Display	'AMD '	414D4420h
Photo CD	'KPCD'	4B504344h
PhotoImageSetter	'imgs'	696D6773h
Gravure	'grav'	67726176h
Offset Lithography	'offs'	6F666673h
Silkscreen	'silk'	73696C6Bh
Flexography	'flex'	666C6578h

**6.4.45 ucrbgTag**

Tag Type: ucrbgType

Tag Signature: 'bfd' (62666420h)

Under color removal and black generation specification. This tag contains curve information for both under color removal and black generation in addition to a general description. The content of this structure is described in 6.5.20.

This tag provides descriptive information only and is not involved in the processing model.

**6.4.46 viewingCondDescTag**

Tag Type: textDescriptionType

Tag Signature: 'vued' (76756564h)

Structure containing invariant and localizable versions of the viewing conditions. The content of this structure is described in 6.5.17.

**6.4.47 viewingConditionsTag**

Tag Type: viewingConditionsType

Tag Signature: 'view' (76696577h)

Viewing conditions parameters. The content of this structure is described in 6.5.25.

## 6.5 Tag type definitions

This clause specifies the type and structure definitions used to create all of the individual tagged elements in the ICC Profile Format. The data type description identifiers are indicated at the right margin of each data or structure definition. An effort was made to make sure one-byte, two-byte and four-byte data lies on one-byte, two-byte and four-byte boundaries respectively. This required occasionally including extra spaces indicated with “reserved for padding” in some tag type definitions. Value 0 is defined to be of “unknown value” for all enumerated data structures.

All tags, including private tags, have as their first four bytes (0..3) a tag signature (a 4-byte sequence) to identify to profile readers what kind of data is contained within a tag. This encourages tag type reuse and allows profile parsers to reuse code when tags use common tag types. The second four bytes (4..7) are reserved for future expansion and must be set to 0 in this version of the specification. Each new tag signature and tag type signature must be registered with the International Color Consortium (see 0.7) in order to prevent signature collisions.

Where not specified otherwise, the least-significant 16 bits of all 32-bit flags in the type descriptions below are reserved for use by the International Color Consortium.

When 7-bit ASCII text representation is specified in types below, each individual character is encoded in 8 bits with the most-significant bit set to zero. The details are presented in 5.3.11.

### 6.5.1 chromaticityType

The chromaticityType information provides basic chromaticity data and type of phosphors or colorants of a monitor to applications and utilities. The byte stream is given below..

**Table 34 — chromaticityType encoding**

Byte Offset	Content	Encoded as...
0..3	'chrm' (6368726Dh) type signature	
4..7	reserved, must be set to 0	
8..9	Number of Device Channels	uInt16Number
10..11	encoded value of phosphor or colorant type	see below
12..19	CIE xy coordinate values of channel 1	u16Fixed16Number[2]
20..27	CIE xy coordinate values of channel 2	u16Fixed16Number[2]
28..35	CIE xy coordinate values of channel 3	u16Fixed16Number[2]
36..end	CIE xy coordinate values of other channels (if needed)	u16Fixed16Number[]

When using this type, it is necessary to assign each color space component to device channel. Table 48 in 6.5.7 shows these assignments.



The encoding for the phosphor or colorant type field is such that:

**Table 35 — Phosphor or colorant encoding**

Phosphor or Colorant Type	Encoded Value	Channel 1 x,y	Channel 2 x,y	Channel 3 x,y
unknown	0000h	any	any	any
ITU-R BT.709	0001h	(0.640, 0.330)	(0.300, 0.600)	(0.150, 0.060)
SMPTE RP145-1994	0002h	(0.630, 0.340)	(0.310, 0.595)	(0.155, 0.070)
EBU Tech.3213-E	0003h	(0.64, 0.33)	(0.29, 0.60)	(0.15, 0.06)
P22	0004h	(0.625, 0.340)	(0.280, 0.605)	(0.155, 0.070)

When the encoded value is 0000h, the actual set of chromaticity values shall be described. Otherwise, the chromaticity values shall match the table values for the given phosphor type.

### 6.5.2 crdInfoType

This type contains the PostScript product name to which this profile corresponds and the names of the companion CRDs. Recall that a single profile can generate multiple CRDs.

**Table 36 — crdInfoType encoding**

Byte Offset	Content	Encoded as...
0..3	'crdi' (63726469h) type signature	
4..7	reserved, must be set to 0	
8..11	PostScript product name character count, including terminating null	ulnt32Number
12..m-1	PostScript product name string	7-bit ASCII
m..m+3	Rendering intent 0 CRD character count, including terminating null	ulnt32Number
m+4 ..n-1	Rendering intent 0 CRD name string	7-bit ASCII
n..n+3	Rendering intent 1 CRD character count, including terminating null	ulnt32Number
n+4..p-1	Rendering intent 1 CRD name string	7-bit ASCII
p..p+3	Rendering intent 2 CRD character count, including terminating null	ulnt32Number
p+4..q-1	Rendering intent 2 CRD name string	7-bit ASCII
q..q+3	Rendering intent 3 CRD character count, including terminating null	ulnt32Number
q+4..r	Rendering intent 3 CRD name string	7-bit ASCII

If a companion CRD is not available for a given profile, then the character count field is zero and there is no string.

See C.1 for more information.

Note: It has been found that `crdInfoType` can contain misaligned data (see 4.1 for the definition of “aligned”). Because the CRD character counts immediately follow variable-length ASCII strings, their alignment is not correct when the length of any of the preceding strings is not a multiple of four. Profile reading and writing software must be written carefully in order to handle these alignment problems.

### 6.5.3 `curveType`

The `curveType` contains a 4-byte count value and a one-dimensional table of 2-byte values. The byte stream is given below.

**Table 37 — `curveType` encoding**

Byte Offset	Content	Encoded as...
0..3	'curv' (63757276h) type signature	
4..7	reserved, must be set to 0	
8..11	count value specifying number of entries that follow	uint32Number
12..end	actual curve values starting with the zeroth entry and ending with the entry <i>count</i> -1.	uint16Number[]

The count value specifies the number of entries in the curve table except as follows:

when *count* is 0, then a linear response (slope equal to 1.0) is assumed,

when *count* is 1, then the data entry is interpreted as a simple gamma value encoded as a `u8Fixed8Number`. Gamma is interpreted canonically and not as an inverse.

Otherwise, the 16-bit unsigned integers in the range 0 to 65535 linearly map to curve values in the interval [0.0, 1.0].

### 6.5.4 `dataType`

The `dataType` is a simple data structure that contains either 7-bit ASCII or binary data, i.e. `textType` data or transparent 8-bit bytes. The length of the string is obtained by subtracting 12 from the element size portion of the tag itself. If this type is used for ASCII data, it must be terminated with a 00h byte.

**Table 38 — `dataType` encoding**

Byte Offset	Content
0..3	'data' (64617461h) type signature
4..7	reserved, must be set to 0
8..11	data flag, 00000000h represents ASCII data, 00000001h represents binary data, other values are reserved for future use
12..end	a string of (element size - 12) ASCII characters or (element size - 12) bytes

### 6.5.5 dateTimeType

This dateTimeType is a 12-byte value representation of the time and date. The actual values are encoded as a dateTimeNumber described in 5.3.1.

**Table 39 — dateTimeType encoding**

Byte Offset	Content	Encoded as...
0..3	'dtim' (6474696Dh) type signature	
4..7	reserved, must be set to 0	
8..19	date and time	dateTimeNumber

### 6.5.6 deviceSettingsType

This type is an array of structures each of which contains platform-specific information about the settings of the device for which this profile is valid.

**Table 40 — deviceSettingsType encoding**

Byte Offset	Content	Encoded as...
0..3	'devs' (64657673h) type signature	
4..7	reserved, must be set to 0	
8..11	count value specifying number of platform entry structures	uInt32Number
12..end	<i>count</i> platform entry structures	see below

Each platform entry structure has the following format:

**Table 41 — Platform encoding**

Byte Offset	Content	Encoded as...
0..3	platform ID signature (one of the Primary Platform signatures from Table 15)	
4..7	size of this structure (including all of its substructures) in bytes	uInt32Number
8..11	count value specifying number of setting combinations structures	uInt32Number
12..end	<i>count</i> setting combinations structures	see below

Each setting combinations structure has the following format:

**Table 42 — Device settings combinations structure**

Byte Offset	Content	Encoded as...
0..3	size of this structure (including all of its substructures) in bytes	uInt32Number
4..7	count value specifying number of setting structures	uInt32Number
8..n	<i>count</i> setting structures	see below

Each setting structure has the following format:

**Table 43 — Device settings setting structure**

Byte Offset	Content	Encoded as...
0..3	setting ID signature	see below
4..7	size (in bytes) per setting value	uInt32Number
8..11	count value specifying number of setting values	uInt32Number
12..n	<i>count</i> setting values	see below

Setting ID signatures are specific to the enclosing platform ID. More settings can be added in the future by the ICC.

The currently defined setting ID signatures and values for the 'MSFT' (Microsoft) platform are encoded as follows:

**Table 44 — Device settings ID signatures**

Setting	Signature	Hex Encoding	Encoded as...
Resolution: X resolution (dpi) in the least-significant 32 bits, Y resolution (dpi) in the most-significant 32 bits	'rsln'	72736C6Eh	uInt64Number (8 bytes per value)
Media type ( <b>see below</b> )	'mtyp'	6D747970h	uInt32Number (4 bytes per value)
Halftone ( <b>see below</b> )	'hftn'	6866746Eh	uInt32Number (4 bytes per value)

Media type values for the 'MSFT' (Microsoft) platform are defined as follows:

**Table 45 — Device settings media type encoding**

<b>Media Type</b>	<b>Encoded Value</b>
DMMEDIA_STANDARD (Standard paper)	1
DMMEDIA_TRANSPARENCY (Transparency)	2
DMMEDIA_GLOSSY (Glossy paper)	3
DMMEDIA_USER (Device-specific media are >= 256)	256

Halftone values for the 'MSFT' (Microsoft) platform are defined as follows:

**Table 46 — Device settings halftone encoding**

<b>Halftone</b>	<b>Encoded Value</b>
DMDITHER_NONE (No dithering)	1
DMDITHER_COARSE (Dither with a coarse brush)	2
DMDITHER_FINE (Dither with a fine brush)	3
DMDITHER_LINEART (LineArt dithering)	4
DMDITHER_ERRORDIFFUSION (LineArt dithering)	5
DMDITHER_RESERVED6 (LineArt dithering)	6
DMDITHER_RESERVED7 (LineArt dithering)	7
DMDITHER_RESERVED8 (LineArt dithering)	8
DMDITHER_RESERVED9 (LineArt dithering)	9
DMDITHER_GRAYSCALE (Device does grayscale)	10
DMDITHER_USER (Device-specific halftones are >= 256)	256

### 6.5.7 lut16Type

This structure converts an input color into an output color using tables with 16-bit precision. This type contains four processing elements: a 3 by 3 matrix (only used when the input color space is XYZ), a set of one dimensional input lookup tables, a multidimensional lookup table, and a set of one dimensional output tables. Data is processed using these elements via the following sequence:

(matrix) ⇒ (1d input tables) ⇒ (multidimensional lookup table) ⇒ (1d output tables).

**Table 47 — lut16Type encoding**

Byte Offset	Content	Encoded as...
0..3	'mft2' (6D667432h) [multi-function table with 2-byte precision] type signature	
4..7	reserved, must be set to 0	
8	Number of Input Channels	uInt8Number
9	Number of Output Channels	uInt8Number
10	Number of CLUT grid points (identical for each side)	uInt8Number
11	Reserved for padding (required to be 00h)	
12..15	Encoded e00 parameter	s15Fixed16Number
16..19	Encoded e01 parameter	s15Fixed16Number
20..23	Encoded e02 parameter	s15Fixed16Number
24..27	Encoded e10 parameter	s15Fixed16Number
28..31	Encoded e11 parameter	s15Fixed16Number
32..35	Encoded e12 parameter	s15Fixed16Number
36..39	Encoded e20 parameter	s15Fixed16Number
40..43	Encoded e21 parameter	s15Fixed16Number
44..47	Encoded e22 parameter	s15Fixed16Number
48..49	Number of input table entries	uInt16Number
50..51	Number of output table entries	uInt16Number
52..n	Input tables	uInt16Number[]
n+1..m	CLUT values	uInt16Number[]
m+1..o	Output tables	uInt16Number[]

The input, output, and grid tables contained in a lut16Type each embodies a one- or multi-dimensional function which maps an input value in the "domain" of the function to an output value in the "range" of the function.

The domain of each of these tables is defined to consist of all real numbers between 0.0 and 65535.0, inclusive. The first entry is located at 0.0, the last entry at 65535.0, and intermediate entries are uniformly spaced using an increment of  $65535.0/(M-1)$ . For the input and output tables, M is the number of entries in the table. For the CLUT, M is the number of grid points along each dimension. Note that since the increment of  $65535.0/(M-1)$  is not necessarily an integer, the domain is specified to be over the real numbers rather than restricting it to the integers only.

The range of a function used to generate the contents of a table is likewise defined to be all real numbers between 0.0 and 65535.0, inclusive. Because the contents of a table are encoded using 16 bits of precision, it is necessary to round each real value to the nearest 16-bit integer.

This means that both the domain and range of the functions represented by the elements of the lut16Type as a whole are all real numbers between 0.0 and 65535.0, inclusive. In many situations it is necessary to convert between these 16-bit values and some other bit precision.

See Annex A for additional guidance on this topic.

The color space used on the PCS side of a lut16Type tag (which may be either the input or output space, or both in the case of an abstract profile) is identified by the Profile Connection Space field in the profile header (see 6.1.6). This field does not distinguish between 8-bit and 16-bit PCS encodings. For the lut16Type tag, the 'Lab' signature is defined to specify the 16-bit CIELAB encoding and the 'XYZ' signature is defined to specify the 16-bit XYZ encoding. Note that this definition only applies to the encoding used as the Profile Connection Space side of the tag. It does NOT apply when these signatures are used on the "Color Space of Data" field in the profile header (see 6.1.5), except in the case of an abstract profile.

The matrix is organized as a 3 by 3 array. The dimension corresponding to the matrix rows varies least rapidly and the dimension corresponding to the matrix columns varies most rapidly and is shown in matrix form below.

$$\begin{bmatrix} e00 & e01 & e02 \\ e10 & e11 & e12 \\ e20 & e21 & e22 \end{bmatrix}$$

When using the matrix of an output profile, and the input data is XYZ, we have

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} e00 & e01 & e02 \\ e10 & e11 & e12 \\ e20 & e21 & e22 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (17)$$

Each input X, Y or Z is an unsigned 1.15 number and each matrix entry is a signed 15.16 number. Therefore, each multiplication in the matrix multiply is 1.15 \* s15.16 = s16.31 and the final sum is also s16.31 (48 bits). From this sum we take bits 31..16 as the unsigned integer result for X', Y', or Z'. These are then used as the inputs to the input tables of the multidimensional LUT. This normalization is used since the number of fractional bits in the input data must be maintained by the matrix operation.

The matrix is mandated to be an identity matrix unless the input is in the XYZ color space.

The input tables are arrays of 16-bit unsigned values. Each input table consists of a minimum of two and a maximum of 4096 two-byte integers. Each input table entry is appropriately normalized to the range 0-65535. The inputTable is of size (InputChannels \* inputTableEntries \* 2) bytes. When stored in this tag, the one-dimensional lookup tables are assumed to be packed one after another in the order described below.

The CLUT is organized as an n-dimensional array with a given number of grid points in each dimension, where n is the number of input channels (input tables) in the transform. The dimension corresponding to the first input channel varies least rapidly and the dimension corresponding to the last input channel varies most rapidly. Each grid point value contains m two-byte integers, where m is the number of output channels. The first sequential two-byte integer of the entry contains the function value for the first output function, the second sequential two-byte integer of the entry contains the function value for the second output function, and so on until all the output functions have been supplied. Each two-byte integer in the CLUT is appropriately normalized to the range of 0 - 65535. The equation for computing the byte size of the CLUT is:

$$CLUTSize = GridPoints^{InputChannels} \cdot OutputChannels \cdot 2 \quad (18)$$

The output tables are arrays of 16-bit unsigned values. Each output table consists of a minimum of two and a maximum of 4096 two-byte integers. Each output table entry is appropriately normalized to the range 0 - 65535. The outputTable is of size (OutputChannels \* outputTableEntries \* 2) bytes. When stored in this tag, the one-dimensional lookup tables are assumed to be packed one after another in the order described in the following paragraph.

When using this type, it is necessary to assign each color space component to an input and output channel. The following table shows these assignments. The channels are numbered according to the order in which their table occurs. Note that additional color spaces can be added simply by defining the signature, channel assignments, and creating the tables.

**Table 48 — lut16Type channel encodings**

Color Space	Channel 1	Channel 2	Channel 3	Channel 4
'XYZ'	X	Y	Z	
'Lab'	L	a	b	
'Luv'	L	u	v	
'YCbCr'	Y	Cb	Cr	
'Yxy'	Y	x	y	
'RGB'	R	G	B	
'GRAY'	K			
'HSV'	H	S	V	
'HLS'	H	L	S	
'CMYK'	C	M	Y	K
'CMY'	C	M	Y	
'2CLR'	Ch. 1	Ch. 2		
'3CLR'	Ch. 1	Ch. 2	Ch. 3	
'4CLR'	Ch. 1	Ch. 2	Ch. 3	Ch. 4

### 6.5.8 lut8Type

This structure converts an input color into an output color using tables of 8-bit precision. This type contains four processing elements: a 3 by 3 matrix (only used when the input color space is XYZ), a set of one dimensional input lookup tables, a multidimensional lookup table, and a set of one dimensional output tables. Data is processed using these elements via the following sequence:



(matrix) ⇒ (1d input tables) ⇒ (multidimensional lookup table) ⇒ (1d output tables).

**Table 49 — lut8Type encoding**

Byte Offset	Content	Encoded as...
0..3	'mft1' (6D667431h) [multi-function table with 1-byte precision] type signature	
4..7	reserved, must be set to 0	
8	Number of Input Channels	uInt8Number
9	Number of Output Channels	uInt8Number
10	Number of CLUT grid points (identical for each side)	uInt8Number
11	Reserved for padding (fill with 00h)	
12..15	Encoded e00 parameter	s15Fixed16Number
16..19	Encoded e01 parameter	s15Fixed16Number
20..23	Encoded e02 parameter	s15Fixed16Number
24..27	Encoded e10 parameter	s15Fixed16Number
28..31	Encoded e11 parameter	s15Fixed16Number
32..35	Encoded e12 parameter	s15Fixed16Number
36..39	Encoded e20 parameter	s15Fixed16Number
40..43	Encoded e21 parameter	s15Fixed16Number
44..47	Encoded e22 parameter	s15Fixed16Number
48..m	Input tables	uInt8Number[]
m+1..n	CLUT values	uInt8Number[]
n+1..o	Output tables	uInt8Number[]

The input, output, and grid tables contained in a lut8Type each embodies a one- or multi-dimensional function which maps an input value in the "domain" of the function to an output value in the "range" of the function.

The domain of each of these tables is defined to consist of all real numbers between 0.0 and 255.0, inclusive. The first entry is located at 0.0, the last entry at 255.0, and intermediate entries are uniformly spaced using an increment of  $255.0/(M-1)$ . For the input and output tables, M is 256. For the CLUT, M is the number of grid points along each dimension. Note that since the increment of  $255.0/(M-1)$  is not necessarily an integer, the domain is specified to be over the real numbers rather than restricting it to the integers only. The range of a function used to generate the contents of a table is likewise defined to be all real numbers between 0.0 and 255.0, inclusive.

Because the contents of a table are encoded using 8 bits of precision, it is necessary to round each real value to the nearest 8-bit integer. This means that both the domain and range of the functions represented by the elements of the lut8Type as a whole are all real numbers between 0.0 and 255.0, inclusive. In many situations it is necessary to convert between these 8-bit values and some other bit precision.

See Annex A for additional guidance on this topic.

The color space used on the PCS side of a lut8Type tag (which may be either the input or output space, or both in the case of an abstract profile) is identified by the Profile Connection Space field in the profile header (see 6.1.6). This field does not distinguish between 8-bit and 16-bit PCS encodings. For the lut8Type tag, the 'Lab' signature is defined to specify the 8-bit CIELAB encoding. Note that this definition only applies to the encoding used as the Profile Connection Space side of the tag. It does NOT apply when these signatures are used on the "Color Space of Data" field in the profile header (see 6.1.5), except in the case of an abstract profile.

An 8-bit XYZ PCS has not been defined, so the interpretation of a lut8Type in a profile that uses the XYZ PCS is implementation specific. Because of the resulting ambiguity and because an 8-bit linear quantization of XYZ results in poor quality, it is recommended that the lut8Type tag not be used in profiles that employ the XYZ PCS.

The matrix is organized as a 3 by 3 array. The dimension corresponding to the matrix rows varies least rapidly and the dimension corresponding to the matrix columns varies most rapidly and is shown in matrix form below.

$$\begin{bmatrix} e00 & e01 & e02 \\ e10 & e11 & e12 \\ e20 & e21 & e22 \end{bmatrix}$$

When using the matrix of an output profile, and the input data is XYZ, we have

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} e00 & e01 & e02 \\ e10 & e11 & e12 \\ e20 & e21 & e22 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (19)$$

Each input X, Y or Z is an unsigned 1.15 number and each matrix entry is a signed 15.16 number. Therefore, each multiplication in the matrix multiply is 1.15 \* s15.16 = s16.31 and the final sum is also s16.31 (48 bits). From this sum we take bits 31..16 as the unsigned integer result for X', Y', or Z'. These are then scaled to the range 0-255 and used as the inputs to the input tables of the multidimensional LUT. This normalization is used since the number of fractional bits in the input data must be maintained by the matrix operation.

The matrix is mandated to be an identity matrix unless the input is in the XYZ color space.

The input tables are arrays of 8-bit unsigned values. Each input table consists of 256 one-byte integers. Each input table entry is appropriately normalized to the range 0-255. The inputTable is of size (InputChannels \* 256) bytes. When stored in this tag, the one-dimensional lookup tables are assumed to be packed one after another in the order described below.

The CLUT is organized as an n-dimensional array with a given number of grid points in each dimension, where n is the number of input channels (input tables) in the transform. The dimension corresponding to the first input channel varies least rapidly and the dimension corresponding to the last input channel varies most rapidly. Each grid point value is an m-byte array, where m is the number of output channels. The first sequential byte of the entry contains the function value for the first output function, the second sequential byte of the entry contains the function value for the second output function, and so on until all the output functions have been supplied. Each byte in the CLUT is appropriately normalized to the range 0 - 255. The equation for computing the byte size of the CLUT is:

$$CLUTSize = GridPoints^{InputChannels} \cdot OutputChannels \quad (20)$$

The output tables are arrays of 8-bit unsigned values. Each output table consists of 256 one-byte integers. Each output table entry is appropriately normalized to the range 0 - 255. The outputTable is of size (OutputChannels \* 256) bytes. When stored in this tag, the one-dimensional lookup tables are assumed to be packed one after another in the order described in the following paragraph.

When using this type, it is necessary to assign each color space component to an input and output channel. The following table shows these assignments. The channels are numbered according to the order in which their table occurs. Note that additional color spaces can be added simply by defining the signature, channel assignments, and creating the tables.

**Table 50 — lut8Type channel encodings**

Color Space	Channel 1	Channel 2	Channel 3	Channel 4
'XYZ'	X	Y	Z	
'Lab'	L	a	b	
'Luv'	L	u	v	
'Yxy'	Y	x	y	
'YCbCr'	Y	Cb	Cr	
'RGB'	R	G	B	
'GRAY'	K			
'HSV'	H	S	V	
'HLS'	H	L	S	
'CMYK'	C	M	Y	K
'CMY'	C	M	Y	
'2CLR'	Ch. 1	Ch. 2		
'3CLR'	Ch. 1	Ch. 2	Ch. 3	
'4CLR'	Ch. 1	Ch. 2	Ch. 3	Ch. 4

### 6.5.9 measurementType

The measurementType information refers only to the internal profile data and is meant to provide profile makers an alternative to the default measurement specifications.

**Table 51 — measurementType structure**

Byte Offset	Content	Encoded as...
0..3	'meas' (6D656173h) type signature	
4..7	reserved, must be set to 0	
8..11	encoded value for standard observer	see below
12..23	XYZ tristimulus values for measurement backing	XYZNumber
24..27	encoded value for measurement geometry	see below
28..31	encoded value for measurement flare	see below
32..35	encoded value for standard illuminant	see below

The encoding for the standard observer field is such that:

**Table 52 — Standard observer encodings**

Standard Observer	Encoded Value
unknown	00000000h
1931 2 degree Observer	00000001h
1964 10 degree Observer	00000002h

The encoding for the measurement geometry field is such that:

**Table 53 — Measurement geometry encodings**

Geometry	Encoded Value
unknown	00000000h
0/45 or 45/0	00000001h
0/d or d/0	00000002h

The encoding for the measurement flare value is shown below and is equivalent to the basic numeric type `u16Fixed16Number` in 5.3.4.

**Table 54 — Measurement flare encodings**

Flare	Encoded Value
0 (0%)	00000000h
1.0 (or 100%)	00010000h

The encoding for the standard illuminant field is such that:

**Table 55 — Standard illuminant encodings**

Standard Illuminant	Encoded Value
unknown	00000000h
D50	00000001h
D65	00000002h
D93	00000003h
F2	00000004h
D55	00000005h
A	00000006h
Equi-Power (E)	00000007h
F8	00000008h

### 6.5.10 namedColorType

NOTE: This type is obsolete, and should not be used in new profiles. Use namedColor2Type instead.

The namedColorType is a count value and array of structures that provide color coordinates for 7-bit ASCII color names. This provides users the ability to create a logo color dictionary between a named color set and a space color specification. The color space is identified by the “color space of data” field of the profile header. In order to maintain maximum portability it is strongly recommended that special characters of the 7-bit ASCII set not be used.

**Table 56 — namedColorType encoding**

Byte Offset	Content	Encoded as...
0..3	'ncol' (6E636F6Ch) type signature	
4..7	reserved, must be set to 0	
8..11	vendor specific flag (least-significant 16 bits reserved for ICC use)	
12..15	count of named colors	ulInt32Number
16..t	prefix for each color name (maximum of 32 bytes) 7-bit ASCII, null terminated	
t+1..u	suffix for each color name (maximum of 32 bytes) 7-bit ASCII, null terminated	
u+1..v	first color root name (maximum of 32 bytes) 7-bit ASCII, null terminated	
v+1..w	first name's color coordinates. Color space of data	
w+1..x	second color root name (maximum of 32 bytes) 7-bit ASCII, null terminated	
x+1..y	second name's color coordinates. Color space of data	
y+1..z	the remaining <i>count-2</i> name structures as described in the first two name structures (assuming <i>count</i> > 2)	

### 6.5.11 namedColor2Type

The namedColor2Type is a count value and array of structures that provide color coordinates for 7-bit ASCII color names. For each named color, a PCS and optional device representation of the color are given. Both representations are 16-bit values. The device representation corresponds to the header's “color space of data” field. This representation should be consistent with the “number of device components” field in the namedColor2Type. If this field is 0, device coordinates are not provided. The PCS representation corresponds to the header's PCS field. The PCS representation is always provided. Color names are fixed-length, 32-byte fields including null termination. In order to maintain maximum portability, it is strongly recommended that special characters of the 7-bit ASCII set not be used.

The namedColorType should no longer be used.

**Table 57 — namedColor2Type encoding**

Byte Offset	Content	Encoded as...
0..3	'ncl2' (6E636C32h) type signature	
4..7	reserved, must be set to 0	
8..11	vendor specific flag (least-significant 16 bits reserved for ICC use)	
12..15	count of named colors	uInt32Number
16..19	number of device coordinates for each named color	uInt32Number
20..51	prefix for each color name (32-byte field including null termination)	7-bit ASCII
52..83	suffix for each color name (32-byte field including null termination)	7-bit ASCII
84..115	first color root name (32-byte field including null termination)	7-bit ASCII
116..121	first named color's PCS coordinates. The encoding is the same as the encodings for the PCS color spaces as described in Annex A. Only 16-bit L*a*b* and XYZ are allowed. The number of coordinates is consistent with the header's PCS.	uInt16Number[]
122..y	first named color's device coordinates. For each coordinate, 0000h represents the minimum value for the device coordinate and FFFFh represents the maximum value for the device coordinate. The number of coordinates is given by the "number of device coordinates" field. If the "number of device coordinates" field is 0, this field is not given.	uInt16Number[]
y+1..z	if <i>count</i> > 1 the remaining <i>count</i> -1 colors are described in a manner consistent with the first named color.	

### 6.5.12 profileSequenceDescType

This type is an array of structures, each of which contains information from the header fields and tags from the original profiles which were combined to create the final profile. The order of the structures is the order

in which the profiles were combined and includes a structure for the final profile. This provides a description of the profile sequence from source to destination, typically used with the DeviceLink profile.

**Table 58 — profileSequenceDescType structure**

Byte Offset	Content
0..3	'pseq' (70736571h) type signature
4..7	reserved, must be set to 0
8..11	count value specifying number of description structures in the array
12..end	<i>count</i> profile description structures

Each profile description structure has the format:

**Table 59 — Profile Description structure**

Byte Offset	Content
0..3	Device manufacturer signature (from corresponding profile's header)
4..7	Device model signature (from corresponding profile's header)
8..15	Device attributes (from corresponding profile's header)
16..19	Device technology information such as CRT, Dye Sublimation, etc. (corresponding profile's technology signature)
20..m	displayable description of device manufacturer (corresponding profile's deviceMfgDescTag)
m+1..n	displayable description of device model (corresponding profile's deviceModelDescTag)

If the deviceMfgDescTag and/or deviceModelDescTag is not present in a component profile, then a "placeholder" tag should be inserted. This tag should have a 1 in the ASCII count field and a terminating null in the ASCII invariant profile description and zeros in the UniCode and ScriptCode count and code fields.

Also note that the entire tag, including the tag type, should be stored.

If the technologyTag is not present, bytes 16..19 should be 00000000h.

### 6.5.13 responseCurveSet16Type

ICC profiles for display and output devices will produce the desired color only while the device has a particular relationship between normalized device codes and physical colorant amount (the reference response). If the response of the device changes (the current response), the profile will no longer produce the correct result. In many cases it is impractical to produce a new profile for the current response, but the change can be compensated for by modifying the single channel device codes.

The purpose of this tag type is to provide a mechanism to relate physical colorant amounts with the normalized device codes produced by lut8Type or lut16Type tags so that corrections can be made for variation in the device without having to produce a new profile. The mechanism can be used by applications to allow users with relatively inexpensive and readily available instrumentation to apply corrections to individual output color channels in order to achieve consistent results.

Two pieces of information are necessary for this compensation: the reference response and the current response. This tag type provides a mechanism that allows applications that create profiles to specify the reference response. The way in which applications determine and make use of the current response is not specified at this time.

The measurements are of the standard variety used in the photographic, graphic arts, and television industries for process control. The measurements are intended to represent colorant amounts and so different measurement techniques are appropriate for different device types.

It is the job of the profile creator to provide reference response data in as many measurement units as practical and appropriate so that applications may select the same units that are measured by the user's instrument. Since it is not possible in general to translate between measurement units, and since most instruments only measure in one unit, providing a wide range of measurement units is vital. The profile originator must decide which measurement units are appropriate for the device.

Here are some examples of suitable measurement units: For process colors, density should be reported. Red-filter density should be reported for the cyan channel, green-filter for the magenta channel, blue-filter for the yellow channel, and visual for the black channel. For other colorants, such as Spot colors or Hi-Fi colors, it is the responsibility of the profile creator to select the appropriate units of measure for the system being profiled. Several different density standards are used around the world, so it is important that profile creators report in as many different density units as possible. Examples of suitable density measurements are: Status T, Status E, Status I and DIN.

This structure relates normalized device codes that would result from a lut16Type tag with density measurements of the resulting colorant amount. Normalized device codes resulting from a lut8Type tag should first be multiplied by 257 (101h).

For those fields that have been structured in arrays of channel data, the channels are ordered as specified for the appropriate color space in Table 48.

**Table 60 — responseCurveSet16Type structure**

Byte Offset	Content	Encoded as...
0..3	'rsc2' (72637332h) [response curve set with 2-byte precision] type signature	
4..7	reserved, must be set to 0	
8..9	number of channels	uInt16Number
10..11	count of measurement types	uInt16Number
12..m	<i>count</i> relative offsets from byte 0 of this structure. Each will point to the response data for the measurement unit.	uInt32Number[]
m+1..n	<i>count</i> response curve structures	see below



Each response curve structure has the format:

**Table 61 — Curve structure**

Byte Offset	Content	Encoded as...
0..3	measurement unit signature	see below
4..m	number of measurements for each channel: 4 = count of chan. 1 measurements, 8 = count of chan. 2 measurements, m-3 = count of final channel measurements	ulInt32Number[]
m+1..n	<i>number-of-channels</i> measurements of patch with the maximum colorant value	XYZNumber[]
n+1..p	<i>number-of-channels</i> response arrays. Each array contains <i>number-of-measurements</i> response16Numbers appropriate to the channel	response16Number[]

Note: The XYZ values are CIE XYZ tristimulus values as described in 5.3.10. The response arrays must be ordered with normalized device code elements increasing.

The measurement unit is encoded as follows:

**Table 62 — Curve measurement encodings**

Measurement Unit	Signature	Hex Encoding
Status A: ANSI PH2.18 densitometer response. This is the accepted standard for reflection densitometers for measuring photographic color prints.	'StaA'	53746141h
Status E: A densitometer response which is the accepted standard in Europe for color reflection densitometers.	'StaE'	53746145h
Status I: A densitometer response commonly referred to as narrow band or interference-type response.	'StaI'	53746149h
Status T: Wide band color reflection densitometer response which is the accepted standard in the United States for color reflection densitometers.	'StaT'	53746154h
Status M: Standard densitometer response for measuring negatives.	'StaM'	5374614Dh
DIN: Measurement according to DIN standard with no polarising filter.	'DN '	444E2020h
DIN with Polarising Filter	'DN P'	444E2050h
Narrow band DIN	'DNN '	444E4E20h
Narrow band DIN with Polarising filter	'DNNP'	444E4E50h

### 6.5.14 s15Fixed16ArrayType

This type represents an array of generic 4-byte/32-bit fixed point quantity. The number of values is determined from the size of the tag.

**Table 63 — s16Fixed16ArrayType encoding**

Byte Offset	Content
0..3	'sf32' (73663332h) type signature
4..7	reserved, must be set to 0
8..end	an array of s15Fixed16Number values

### 6.5.15 screeningType

The screeningType describes various screening parameters including screen frequency, screening angle, and spot shape.

**Table 64 — screeningType structure**

Byte Offset	Content	Encoded as...
0..3	'scrn' (7363726Eh) type signature	
4..7	reserved, must be set to 0	
8..11	screening flag	see below
12..15	number of channels	
16..19	channel #1 frequency	s15Fixed16Number
20..23	channel #1 screen angle (degrees)	s15Fixed16Number
24..27	channel #1 spot shape	see below
28..end	frequency, screen angle, and spot shape for additional channels	

Screening flag encoding is such that:

**Table 65 — Screening flag encodings**

Attribute	Bit Position
Use Printer Default Screens (true is 1)	0
Frequency units in Lines/Inch (value 1) or Lines/cm (value 0)	1

Spot function encoding is such that:

**Table 66 — Screening spot encodings**

Spot Function Value	Encoded Value
unknown	0
printer default	1
round	2
diamond	3
ellipse	4
line	5
square	6
cross	7

### 6.5.16 signatureType

The signatureType contains a four-byte sequence used for signatures. Typically this type is used for tags that need to be registered and can be displayed on many development systems as a sequence of four characters. Sequences of less than four characters are padded at the end with spaces, 20h.

**Table 67 — signatureType encoding**

Byte Offset	Content
0..3	'sig' (73696720h) type signature
4..7	reserved, must be set to 0
8..11	four-byte signature

### 6.5.17 textDescriptionType

The textDescriptionType is a complex structure that contains three types of text description structures: 7-bit ASCII, Unicode and ScriptCode. Since no single standard method for specifying localizable character sets exists across the major platform vendors, including all three provides access for the major operating systems. The 7-bit ASCII description is to be an invariant, nonlocalizable name for consistent reference. It is preferred that both the Unicode and ScriptCode structures be properly localized.

Some applications look for special strings of text in the descriptions. These strings are delimited by << and >> tokens, and need to remain constant. Therefore, avoid translating or otherwise modifying text of the form "<<text>>" when localizing a profile.

The localized Macintosh profile description contains 67 bytes of data, of which at most *count* bytes contain a ScriptCode string, including a null terminator. The count cannot be greater than 67.

The count field for each types are defined as follows:

ASCII: The count is the length of the string in bytes including the null terminator.

Unicode: The count is the number of characters including a Unicode null where a character is always two bytes.

ScriptCode: The count is the length of the string in bytes including the terminating null.

If both Unicode and ScriptCode structures cannot be localized then the following guidelines should be used:

If Unicode is not native on the platform, then the Unicode language code and Unicode count should be filled in as 0, with no data placed in the Unicode localizable profile description area.

If Scriptcode is not native on the platform, then the ScriptCode code and ScriptCode count should be filled in as 0. The 67-byte localizable Macintosh profile description should be filled with 0's.

**Table 68 — textDescriptionType encoding**

Byte Offset	Content	Encoded as...
0..3	'desc' (64657363h) type signature	
4..7	reserved, must be set to 0	
8..11	ASCII invariant description count, including terminating null (description length)	ulnt32Number
12..n-1	ASCII invariant description	7-bit ASCII
n..n+3	Unicode language code	ulnt32Number
n+4..n+7	Unicode localizable description count (description length)	ulnt32Number
n+8..m-1	Unicode localizable description	
m..m+1	ScriptCode code	ulnt16Number
m+2	Localizable Macintosh description count (description length)	ulnt8Number
m+3..m+69	Localizable Macintosh description	

An example of a `textDescriptionType` which has neither Unicode nor `ScriptCode` is:

**Table 69 — `textDescriptionType` encoding example**

Byte Offset	Content
0..3	'desc' (64657363h)
4..7	ulnt32Number = 0 (reserved)
8..11	ulnt32Number = 15 (ASCII count)
12..26	ASCII string "An ICC Profile" with a NULL byte at the end
27..30	ulnt32Number = 0 (Unicode code)
31..34	ulnt32Number = 0 (Unicode count)
35..36	ulnt16Number = 0 ( <code>ScriptCode</code> code)
37	ulnt8Number = 0 ( <code>ScriptCode</code> count)
38..104	ulnt8Number[67] filled with 0's

Note: It has been found that `textDescriptionType` can contain misaligned data (see 4.1 for the definition of "aligned"). Because the Unicode language code and Unicode count immediately follow the ASCII description, their alignment is not correct when the ASCII count is not a multiple of four. The `ScriptCode` code is misaligned when the ASCII count is odd. Profile reading and writing software must be written carefully in order to handle these alignment problems.

#### 6.5.18 `textType`

The `textType` is a simple text structure that contains a 7-bit ASCII text string. The length of the string is obtained by subtracting 8 from the element size portion of the tag itself. This string must be terminated with a 00h byte.

**Table 70 — `textType` encoding**

Byte Offset	Content
0..3	'text' (74657874h) type signature
4..7	reserved, must be set to 0
8..end	a string of (element size - 8) 7-bit ASCII characters

### 6.5.19 u16Fixed16ArrayType

This type represents an array of generic 4-byte/32-bit quantity. The number of values is determined from the size of the tag.

**Table 71 — u16Fixed16ArrayType encoding**

Byte Offset	Content
0..3	'uf32' (75663332h) type signature
4..7	reserved, must be set to 0
8..end	an array of u16Fixed16Number values

### 6.5.20 ucrbgType

This type contains curves representing the under color removal and black generation and a text string which is a general description of the method used for the UCR and BG.

**Table 72 — ucrbgType structure**

Byte Offset	Content	Encoded as...
0..3	'bfd ' (62666420h) type signature	
4..7	reserved, must be set to 0	
8..11	count value specifying number of entries in the UCR curve	ulnt32Number
12..m	actual UCR curve values starting with the zeroth entry and ending with the entry <i>count-1</i> . If the count is 1, the value is a percent.	ulnt16Number[]
m+1..m+4	count value specifying number of entries in the BG curve	ulnt32Number
m+5..n	actual BG curve values starting with the zeroth entry and ending with the entry <i>count-1</i> . If the count is 1, the value is a percent.	ulnt16Number[]
n+1..p	a string of ASCII characters, with a null terminator.	7-bit ASCII

Note: It has been found that ucrbgType can contain misaligned data (see 4.1 for the definition of "aligned"). Because the BG count immediately follows the UCR curve values, its alignment is not correct when the UCR count is odd. Profile reading and writing software must be written carefully in order to handle this alignment problem.

### 6.5.21 ulnt16ArrayType

This type represents an array of generic 2-byte/16-bit quantity. The number of values is determined from the size of the tag.

**Table 73 — ulnt16ArrayType encoding**

Byte Offset	Content
0..3	'ui16' (75693136h) type signature
4..7	reserved, must be set to 0
8..end	an array of unsigned 16-bit integers

### 6.5.22 ulnt32ArrayType

This type represents an array of generic 4-byte/32-bit quantity. The number of values is determined from the size of the tag.

**Table 74 — ulnt32ArrayType encoding**

Byte Offset	Content
0..3	'ui32' (75693332h) type signature
4..7	reserved, must be set to 0
8..end	an array of unsigned 32-bit integers

### 6.5.23 ulnt64ArrayType

This type represents an array of generic 8-byte/64-bit quantity. The number of values is determined from the size of the tag.

**Table 75 — ulnt64ArrayType encoding**

Byte Offset	Content
0..3	'ui64' (75693634h) type signature
4..7	reserved, must be set to 0
8..end	an array of unsigned 64-bit integers

### 6.5.24 ulnt8ArrayType

This type represents an array of generic 1-byte/8-bit quantity. The number of values is determined from the size of the tag.

**Table 76 — ulnt8ArrayType encoding**

Byte Offset	Content
0..3	'ui08' (75693038h) type signature
4..7	reserved, must be set to 0
8..end	an array of unsigned 8-bit integers

### 6.5.25 viewingConditionsType

This type represents a set of viewing condition parameters including: absolute illuminant white point tristimulus values and absolute surround tristimulus values.

**Table 77 — viewingConditionsType encoding**

Byte Offset	Content	Encoded as...
0..3	'view' (76696577h) type signature	
4..7	reserved, must be set to 0	
8..19	absolute XYZ value for illuminant in $\text{cd/m}^2$	XYZNumber
20..31	absolute XYZ value for surround in $\text{cd/m}^2$	XYZNumber
32..35	illuminant type	as described in measurementType

### 6.5.26 XYZType

The XYZType contains an array of three encoded values for the XYZ tristimulus values. The number of sets of values is determined from the size of the tag. The byte stream is given below. Tristimulus values must be non-negative. The signed encoding allows for implementation optimizations by minimizing the number of fixed formats.

**Table 78 — XYZType encoding**

Byte Offset	Content	Encoded as...
0..3	'XYZ ' (58595A20h) type signature	
4..7	reserved, must be set to 0	
8..end	an array of XYZ numbers	XYZNumber



## Annex A Color spaces

The International Color Profile Format supports a variety of both device-dependent and device-independent color spaces divided into three basic families: 1) CIEXYZ based, 2) RGB based, and 3) CMY based.

The CIE color spaces are defined in CIE publication 15.2 on Colorimetry. A subset of the CIEXYZ based spaces are also defined as connection spaces. The device dependent spaces below are only representative and other device dependent color spaces may be used without needing to update the profile format specification or the software that uses it.

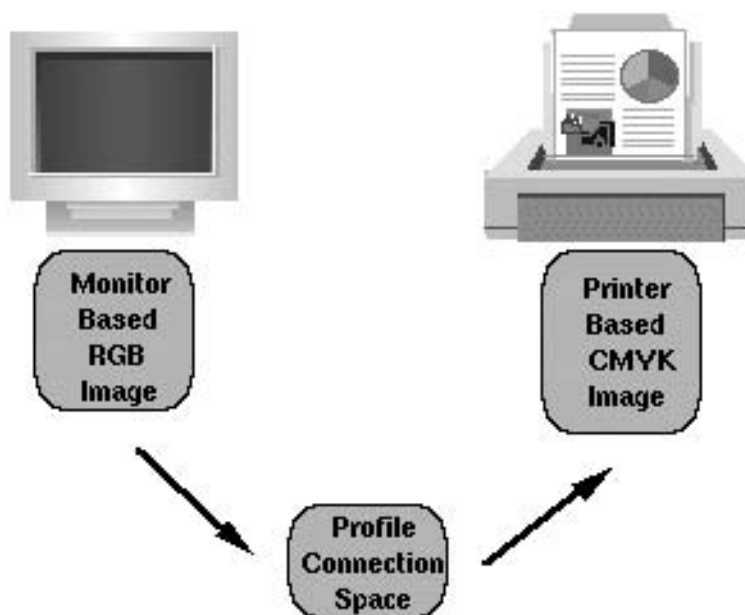
**Table 79 — CIE color spaces**

<b>Base Space</b>	<b>Description</b>	<b>Derivative Space</b>
CIEXYZ	base CIE device-independent color space	CIELAB
GRAY	monochrome device-dependent color space	
RGB	base additive device-dependent color space	HLS, HSV
CMY	base subtractive device-dependent color space	CMYK

### A.1 Profile Connection Spaces

A key component of these profiles is a well-defined profile connection space. This space is the interface which provides an unambiguous connection between the input and output profiles as illustrated in the diagram below. The profile connection space is based on the CIE 1931 standard observer. This experimentally derived standard observer provides a very good representation of the human visual system color matching capabilities. Unlike device dependent color spaces, if two colors have the same CIE colorimetry they will match if viewed under the same conditions. Because the imagery is typically produced for a wide variety of viewing environments, it is necessary to go beyond simple application of the CIE system.

Figure 3 — Profile connection space illustration



The profile connection space is defined as the CIE colorimetry which will produce the desired color appearance if rendered on a reference imaging media and viewed in a reference viewing environment. This reference corresponds to an ideal reflection print viewed in an ANSI standard viewing booth.

The default measurement parameters for the profile connection space and all other color spaces defined in this specification are based on the ANSI CGATS.5-1993 standard, "Graphic technology - Spectral measurement and colorimetric computation for graphic arts images." Essentially this defines a standard illuminant of D50, the 1931 CIE standard observer, and 0/45 or 45/0 reflectance measurement geometry. The reference viewing condition shall be ISO 3664 viewing condition P2 using the recommended 20% surround reflectance. This is a graphics arts and photography print viewing environment with a D50 illumination level of 500 lux.

One of the first steps in profile building involves measuring the colorimetry of a set of colors from some imaging media or display. If the imaging media or viewing environment differ from the reference, it will be necessary to adapt the measured colorimetry to that appropriate for the profile connection space. These adaptations account for such differences as white point chromaticity and luminance relative to an ideal reflector, maximum density, viewing surround, viewing illuminant, and flare. Currently, it is the responsibility of the profile builder to do this adaptation.

However, the possibility of allowing a variable illuminant in the PCS is under active consideration by the International Color Consortium. For this reason, a PCS illuminant field is in the profile header, but must be set to the CIE Illuminant D50 [X=0.9642, Y=1.0000, Z=0.8249].

The PCS is based on media-relative colorimetry. This is in comparison to ICC-absolute colorimetry. In ICC-absolute colorimetry colors are represented with respect to the illuminant, for example D50. In media-relative colorimetry, colors are represented with respect to a combination of the illuminant and the media's white, e.g. unprinted paper. The translation from media-relative colorimetry XYZ data,  $XYZ_r$  to ICC-absolute colorimetric data,  $XYZ_a$ , is given by

$$X_a = \left( \frac{X_{mw}}{X_i} \right) \cdot X_r \quad (\text{A.1})$$

$$Y_a = \left( \frac{Y_{mw}}{Y_i} \right) \cdot Y_r \quad (\text{A.2})$$

$$Z_a = \left( \frac{Z_{mw}}{Z_i} \right) \cdot Z_r \quad (\text{A.3})$$

where  $XYZ_{mw}$  represents the chromatically adapted media white point as found in the mediaWhitePoint-Tag and  $XYZ_i$  represents the illuminant white (which is D50).

The actual media and actual viewing conditions will typically differ from the reference conditions. The profile specification defines tags which provide information about the actual white point and black point of a given media or display. These tags may be used by a CMM to provide functionality beyond that of the default. For example, an advanced CMM could use the tags to adjust colorimetry based on the Dmin of a specific media. A tag is also provided to describe the viewing environment. This information is useful in choosing a profile appropriate for the intended viewing method.

There are many ways of encoding CIE colorimetry. This specification provides three methods in order to satisfy conflicting requirements for accuracy and storage space. These encodings, an 8-bit/component CIELAB encoding, a 16-bit/component CIELAB encoding, and a 16-bit/component CIEXYZ encoding are described below. The CIEXYZ space represents a linear transformation of the derived matching responses and the CIELAB space represents a transformation of the CIEXYZ space into one that is nearly perceptually uniform. This uniformness allows color errors to be equally weighted throughout its domain. While supporting multiple CIE encodings increases the complexity of color management, it provides immense flexibility in addressing different user requirements such as color accuracy and memory footprint.

It is important to understand that the PCS encodings do not represent a quantization of the connection space. The purpose of the encodings is to allow points within the space to be specified. Since the processing models benefit from interpolation between table entries, the interpolated AToB results should be used as the inputs to the BToA transforms. The AToB results should not be rounded to the nearest encoding value.

For the CIEXYZ encoding, each component (X, Y, and Z) is encoded as a fixed unsigned 16-bit quantity which has 15 fractional bits (u1.15).

An example of this encoding is:

**Table 80 — CIEXYZ encoding**

0	0000h
1.0	8000h
$1 + (32767/32768)$	FFFFh

The encoding encompasses a large range of values. However, not all encodable values are useable. Since the PCS represents an ideal reflection print, and the media is a perfect diffuser, the largest valid XYZ values are those of the PCS illuminant (specified in the profile header).<sup>1)</sup> This encoding was chosen to allow for PCS illuminants that have an X or Z greater than 1.0.

For the CIELAB PCS encodings, the L\* values have a different encoding than the a\* and b\* values. The L\* encoding is:

**Table 81 — CIELAB L\* encoding**

Value (L*)	8-bit	16-bit
0	00h	0000h
100.0	FFh	FF00h
100 + (25500/65280)	none	FFFFh

You can convert between the 8-bit and 16-bit encodings by multiplying or dividing by 256.

Although the 16-bit encoding can represent values slightly greater than 100.0, these are not valid PCS L\* values and they should not be used.

The a\* and b\* encoding is:

**Table 82 — CIELAB a\* or b\* encoding**

Value (a* or b*)	8-bit	16-bit
-128.0	00h	0000h
0	80h	8000h
127.0	FFh	FF00h
127 + (255/256)	none	FFFFh

Note that this is not "two's complement" encoding, but a linear scaling after an offset of 128. This encoding was chosen to prevent discontinuities in CLUTs when going from negative to positive values.

You can convert between the 8-bit and 16-bit encodings by multiplying or dividing by 256.

Note that the 16-bit encoding can represent values slightly greater than 127.0. Since a\* and b\* have no defined limits, these are valid PCS values.

Because of the way that PCS encodings map to input tables, the BToAn tags must be able to handle invalid PCS values. However, the results of sending invalid values to these tags is up to the creator of the profile.

An important point to be made is that the PCS is not necessarily intended for the storage of images. A separate series of "interchange color spaces" may be defined in a future version of this specification for this purpose. The design choices made for these spaces (colorimetric encoding, reference media, viewing conditions, etc.) might be different than that of the PCS.

---

<sup>1)</sup> For a D50 illuminant, the largest valid XYZ values are [0.9642, 1.0, 0.8249], or [7B6Bh, 8000h, 6996h] in encoded form. Note that the PCS illuminant values are stored in s15.16 format, so you must translate them to u1.15 format to find the encoded PCS limits.

## A.2 External and internal conversions

CMMs or other applications that use ICC tags to perform color transformations typically need to perform two types of data processing in addition to table interpolation. First, because the color values being processed (such as image pixels) may not match the native precision of an ICC tag (such as a lut16Type or lut8Type), it may be necessary to alter the precision of the input to (or results from) these transforms. Second, because there is more than one PCS encoding, it may be necessary to convert the output from a first transform before applying it to the input of a second transform. These two types of additional processing may be thought of as primarily affecting the **external** and **internal** interfaces of ICC processing, respectively.

In the first (external) case, the appropriate conversion method is to multiply each color value by  $(2^M-1)/(2^N-1)$ , where N is the starting number of bits and M is the required number of bits. This converts a number with values from 0 to  $(2^N-1)$  to a number with values from 0 to  $(2^M-1)$ . For example, to prepare an 8-bit image value for input to a lut16Type tag the scale factor is  $(2^{16}-1)/(2^8-1) = 65535.0/255.0 = 257.0$ . Note that the colors represented by the scaled numbers (be they device coordinates or some other color space) are not altered by the change in precision. For example, if a particular image value represents an L\* of 31.0, then the scaled value also represents an L\* of 31.0. Additionally, if an integer value is required from the scaling operation, it should be obtained via rounding rather than truncation.

In the second (internal) case, the appropriate conversion uses the CIE equations to convert between CIEXYZ and CIELAB. Additionally, special attention should be paid to the colors representable by the 8-bit and 16-bit PCS encodings - they are not the same! For example, conversion from the 8-bit encodings to the 16-bit encodings requires a scale factor of  $\{16\text{-bit } L^* \text{ of } 100\}/\{8\text{-bit } L^* \text{ of } 100\} = 65280.0/255.0 = 256.0$  rather than 257.0 as is appropriate for the image values case described above. This is because both the precision and the representable extremes of the color gamut of the encodings are different.

## A.3 Rendering Intents

In general, actual device gamuts will not be large enough to reproduce the desired color appearances communicated by the PCS values. Four rendering intents (gamut mapping styles) are provided to address this problem. Each one represents a different compromise. The colorimetric rendering intents operate on measurement based colorimetric values as chromatically adapted to the reference viewing environment. This adaptation will usually be indicated in the chromaticAdaptationTag. The other rendering intents operate on colorimetric values which are corrected in an as-needed fashion to account for any differences between devices, media, and viewing conditions.

### A.3.1 Colorimetric Intents

The colorimetric intents preserve the relationships between in-gamut colors at the expense of out-of-gamut colors. Mapping of out-of-gamut colors is not specified but should be consistent with the intended use of the transform.

#### A.3.1.1 MediaWhitePoint Tag

The mediaWhitePointTag contains CIE 1931 XYZ colorimetry of the white point of the actual medium. If the viewer completely adapts to the white point of the medium (as is often the case with monitors) this tag should be set to Xi, Yi, Zi (the PCS white point). If chromatic adaptation is being applied to the PCS values, the adaptation should be applied to the mediaWhitePointTag values as well.

### **A.3.1.2 Media-Relative Colorimetric Intent**

This intent rescales the in-gamut, chromatically adapted tristimulus values such that the white point of the actual medium is mapped to the white point of the reference medium (for either input or output). It is useful for colors that have already been mapped to a medium with a smaller gamut than the reference medium (and therefore need no further compression).

### **A.3.1.3 ICC-Absolute Colorimetric Intent**

For this intent, the chromatically adapted tristimulus values of the in-gamut colors are unchanged. It is useful for spot colors and when simulating one medium on another (proofing). Note that this definition of ICC-absolute colorimetry is actually called “relative colorimetry” in CIE terminology, since the data has been normalized relative to the illuminant.

### **A.3.1.4 Applying the ICC-Absolute Colorimetric Intent**

Profiles do not contain a separate transform for the ICC-absolute colorimetric intent. When this intent is needed, it is generated using the media-relative colorimetric intent and scaling the PCS values by the ratio of the destination profile `mediaWhitePointTag` to the source profile `mediaWhitePointTag` (as described in this annex). This applies to profiles using the TRC/matrix model, which is defined to use the media-relative colorimetric intent.

## **A.3.2 Perceptual Intent**

The exact gamut mapping of the perceptual intent is vendor specific and involves compromises such as trading off preservation of contrast in order to preserve detail throughout the tonal range. It is useful for general reproduction of images, particularly pictorial or photographic-type images.

## **A.3.3 Saturation Intent**

The exact gamut mapping of the saturation intent is vendor specific and involves compromises such as trading off preservation of hue in order to preserve the vividness of pure colors. It is useful for images which contain objects such as charts or diagrams.

## Annex B Embedding Profiles

This annex details the requirements and options for embedding device profiles within PICT, EPS, TIFF, JFIF, and GIF image files. All profiles except Abstract and DeviceLink profiles can be embedded. The complete profile must be embedded with all tags intact and unchanged.

### B.1 Embedding ICC profiles in PICT files

Apple has defined a new QuickDraw picture comment type for embedded ICC profiles. The picture comment value of 224 is followed by a 4-byte selector that describes the type of data in the comment. Using a selector allows the flexibility to embed more CMM related information in the future. The following selectors are currently defined:

**Table 83 — PICT selectors**

Selector	Description	
0	Beginning of an ICC profile.	Profile data to follow.
1	Continuation of ICC profile data.	Profile data to follow.
2	End of ICC profile data.	No profile data follows.

Because the `dataSize` parameter of the `PicComment` procedure is a signed 16-bit value, the maximum amount of profile data that can be embedded in a single picture comment is 32763 bytes (32767 - 4 bytes for the selector). You can embed a larger profile by using multiple picture comments of selector type 1. The profile data must be embedded in consecutive order, and the last piece of profile data must be followed by a picture comment of selector type 2.

All embedded ICC profiles, including those that fit within a single picture comment, must be followed by the end-of-profile picture comment (selector 2), as shown in the following examples.

Example 1: Embedding a 20K profile.

```
PicComment kind = 224, dataSize = 20K + 4, selector = 0, profile data = 20K
```

```
PicComment kind = 224, dataSize = 4, selector = 2
```

Example 2: Embedding a 50K profile.

```
PicComment kind = 224, dataSize = 32K, selector = 0, profile data = 32K - 4
PicComment kind = 224, dataSize = 18K + 8, selector = 1, profile data = 18K + 4
```

```
PicComment kind = 224, dataSize = 4, selector = 2
```

In ColorSync 1.0, picture comment types `CMBeginProfile` (220) and `CMEndProfile` (221) are used to begin and end a picture comment. The `CMBeginProfile` comment is not supported for ICC profiles; however, the `CMEndProfile` comment can be used to end the current profile and begin using the System Profile for both ColorSync 1.0 and 2.0.

The `CMEnableMatching` (222) and `CMDisableMatching` (223) picture comments are used to begin and end color matching in both ColorSync 1.0 and 2.0

See “Advanced Color Imaging on the Mac OS”, Apple Computer 1995, for more information about picture comments.

## B.2 Embedding ICC profiles in EPS files

There are two places within EPS files that embedding International Color Consortium (ICC) profiles are appropriate. 1) Associated with a screen preview. 2) Associated with the page description. Embedding ICC profiles within a screen preview is necessary so that applications using this screen preview to display a representation of the EPS page description can do so with accurate colors. Embedding ICC profiles within a page description is necessary so that sophisticated applications, such as OPI server software, can perform color conversions along with image replacement. For general information concerning PostScript's Document Structuring Conventions (DSC), the EPS file format, or specific PostScript operators, see the PostScript Language Reference Manual, second edition.

1) There are a variety of different methods of storing a screen preview within an EPS file depending on the intended environment. For cross platform applications with embedded ICC profiles, TIFF screen previews are recommended. The TIFF format has been extended to support the embedding of ICC profiles. ICC profiles can also be embedded in a platform specific manner. For example on the Macintosh, Apple has defined a method for embedding ICC profiles in PICT files (see B.1).

Note that a given page description may use multiple distinct color spaces. In such cases, color conversions must be performed to a single color space to associate with the screen preview.

2) ICC profiles can also be embedded in the page description portion of an EPS file using the %%BeginICCPProfile: / %%EndICCPProfile comments. This convention is defined as follows.

```
%%BeginICCPProfile: <profileid> <numberof> [<type> [<bytesorlines>]]
<profileid> ::= <text> (Profile ID)
<numberof> ::= <int> (Lines or physical bytes)
<type> ::= Hex | ASCII (Type of data)
<bytesorlines> ::= Bytes | Lines (Read in bytes or lines)
%%EndICCPProfile (no keywords)
```

These comments are designed to provide information about embedded ICC profiles. If the type argument is missing, ASCII data is assumed. ASCII refers to an ASCII base-85 representation of the data. If the bytesorlines argument is missing, <numberof> shall be considered to indicate bytes of data. If <numberof> = -1, the number of bytes of data are unknown. In this case, to skip over the profile one must read data until the encountering the %%EndICCPProfile comment.

<profileID> provides the profile's ID in order to synchronize it with PostScript's setcolorspace and findcolorrendering operators and associated operands (see below). Note that <numberof> indicates the bytes of physical data, which vary from the bytes of virtual data in some cases. With hex, each byte of virtual data is represented by two ASCII characters (two bytes of physical data). Although the PostScript interpreter ignores white space and percent signs in hex and ASCII data, these count toward the byte count.

Each line of profile data shall begin with a single percent sign followed by a space (%). This makes the entire profile section a PostScript language comment so the file can be sent directly to a printer without modification. The space avoids confusion with the open extension mechanism associated with DSC comments.

ICC profiles can be embedded within EPS files to allow sophisticated applications, such as OPI server software, to extract the profiles, and to perform color processing based on these profiles. In such situations it is desirable to locate the page description's color space and rendering intent, since this color space and rendering intent may need to be modified based on any color processing. The %%BeginSetColorSpace:



/%%EndSetColorSpace and %%BeginRenderingIntent : /%%EndRenderingIntent comments are used to delimit the color space and rendering intent respectively.

```
%%BeginSetColorSpace: <profileid>
<profileid> ::= <text> (ICC Profile ID)
%%EndSetColorSpace (no keywords)
```

<profileid> provides the ICC profile's ID corresponding to this color space. The ICC profile with this profile must have occurred in the PostScript job using the %%BeginICCProfile: /%%EndICCProfile comment convention prior to this particular %%BeginSetColorSpace: comment.

An example usage is shown here for CIE 1931 (XYZ)-space with D65 white point that refers to the ICC profile with <profileid> = XYZProfile.

```
%%BeginSetColorSpace: XYZProfile
[/CIEBasedABC <<
/WhitePoint [0.9505 1 1.0890]
/RangeABC [0 0.9505 0 1 0 1.0890]
/RangeLMN [0 0.9505 0 1 0 1.0890]
>>] setcolorspace
%%EndSetColorSpace
```

Note that the setcolorspace command is included within the comments. The PostScript enclosed in these comments shall not perform any other operations other than setting the color space and shall have no side effects.

```
%%BeginRenderingIntent: <profileid>
<profileid> ::= <text> (ICC Profile ID)
%%EndRenderingIntent (no keywords)
```

<profileid> provides the ICC profile's ID corresponding to this rendering intent. The ICC profile with this profile must have occurred in the PostScript job using the %%BeginICCProfile: /%%EndICCProfile comment convention prior to invocation of this particular %%BeginRenderingIntent: comment.

An example usage is shown here for the Perceptual rendering intent that refers to the ICC profile with <profileid> = RGBProfile.

```
%%BeginRenderingIntent: RGBProfile
/Perceptual findcolorrendering pop
/ColorRendering findresource setcolorrendering
%%EndRenderingIntent
```

Note that the setcolorrendering command is included within the comments. The PostScript enclosed in these comments shall not perform any other operations other than setting the rendering intent and shall have no side effects.

### B.3 Embedding ICC profiles in TIFF files

The discussion below assumes some familiarity with TIFF internal structure. It is beyond the scope of this document to detail the TIFF format, and readers are referred to the "TIFF™ Revision 6.0" specification, which is available from Adobe Systems Incorporated.

The International Color Consortium has been assigned a private TIFF tag for purposes of embedding ICC device profiles within TIFF image files. This is not a required TIFF tag, and Baseline TIFF readers are not currently required to read it. It is, however, strongly recommended that this tag be honored.

An ICC device profile is embedded, in its entirety, as a single TIFF field or Image File Directory (IFD) entry in the IFD containing the corresponding image data. An IFD should contain no more than one embedded profile. A TIFF file may contain more than one image, and so, more than one IFD. Each IFD may have its own embedded profile. Note, however, that Baseline TIFF readers are not required to read any IFDs beyond the first one.

The structure of the ICC Profile IFD Entry is as follows.

**Table 84 — ICC profile IFD entry structure**

Byte Offset	Content
0..1	The TIFFTag that identifies the field = 34675(8773.H)
2..3	The field Type = 7 = UNDEFINED (treated as 8-bit bytes).
4..7	The Count of values = the size of the embedded ICC profile in bytes.
8..11	The Value Offset = the file offset, in bytes, to the beginning of the ICC profile.

Like all IFD entry values, the embedded profile must begin on a 2-byte boundary, so the Value Offset will always be an even number.

A TIFF reader should have no knowledge of the internal structure of an embedded ICC profile and should extract the profile intact.

### B.4 Embedding ICC profiles in JFIF files

The JPEG standard (ISO/IEC 10918-1) supports application specific data segments. These segments may be used for tagging images with ICC profiles. The APP2 marker is used to introduce the tag. Given that there are only 15 supported APP markers, there is a chance of many applications using the same marker. ICC tags are thus identified by beginning the data with a special null terminated byte sequence, "ICC\_PROFILE".

The length field of a JPEG marker is only two bytes long; the length of the length field is included in the total. Hence, the values 0 and 1 are not legal lengths. This would limit maximum data length to 65533. The identification sequence would lower this even further. As it is quite possible for an ICC profile to be longer than this, a mechanism must exist to break the profile into chunks and place each chunk in a separate marker. A mechanism to identify each chunk in sequence order would thus be useful.

The identifier sequence is followed by one byte indicating the sequence number of the chunk (counting starts at 1) and one byte indicating the total number of chunks. All chunks in the sequence must indicate the same total number of chunks. The one-byte chunk count limits the size of embeddable profiles to 16,707,345 bytes.

## **B.5 Embedding ICC profiles in GIF files**

The GIF89a image file format supports Application Extension blocks, which are used for "application specific" information. These blocks may be used for tagging images with ICC profiles.

The Application Identifier for an embedded profile shall be the following 8 bytes: "ICCRGBG1". The Authentication Code shall be "012". The entire profile shall be embedded as application data, using the conventional technique of breaking the data into chunks of at most 255 bytes of data.

## Annex C

### PostScript Level 2 Tags

The PostScript Level 2 tags are provided in order to control exactly the PostScript Level 2 operations that should occur for a given profile. These tags are only valid for PostScript Level 2 (and conceivably future versions of PostScript) devices, and are not generally supported in PostScript Level 1 devices. In addition, some of the tags may correspond to PostScript operations that are not supported in all PostScript Level 2 devices. Using such tags requires first checking for the available operators. All operators described in the PostScript Language Reference Manual, second edition, are available on all PostScript Level 2 devices. Documentation for extensions to PostScript Level 2 are available through Adobe's Developer Support Organization. In addition, guidelines for PostScript compatibility with this profile format are available. For details of such operator support, compatibility guidelines, the PostScript Level 2 device independent color model, or other PostScript-related issues contact Adobe's Developer Support Organization.

In general, there is a straightforward relationship between the profile's header fields and tags, and these PostScript tags. It is anticipated that the various CMSs that support this profile format will also provide support for these optional PostScript tags. To verify such support contact the CMS vendors directly. In cases where such support is provided, and the desired model of operations is the same for PostScript processing as it is for CMS processing, these tags can be omitted, since all necessary information is in the profile itself. In the case where such CMS support is in question or processing different than that provided by an arbitrary CMS is desired, these tags can be populated to provide exact control over the PostScript processing. For example, if private tags are used in the profile to achieve a non-public type of processing on certain CMSs, such processing can be achieved on a PostScript device by populating the appropriate PostScript tags.

Some of the PostScript tags have a tag type of `textType` or `ulnt8Type`. This choice is provided in order to match the properties of the communications channel to the data in these tags. Encoding the data in `ulnt8Type` form is recommended to save memory and/or reduce transmission times. Applications and drivers may convert it to ASCII Coded PostScript, Binary Coded PostScript, or Token Binary Coded PostScript or leave it in binary format to match the requirements of the communications channel. Applications and drivers are responsible for this potential conversion from binary data to channel compatible data. The data should be encoded in `textType` in those cases where the amount of data is relatively small or where the conversion from binary to channel compatible data is not available.

The PostScript contained in these tags is not self evaluating - it simply provides operands. These operands must be followed by operators like `setcolorspace`, `setcolorrendering`, and `findcolorrendering`.

#### C.1 Synchronizing profiles and CRDs

ICC profiles (profiles) on the host and PostScript color rendering dictionaries (CRDs) in the printer can contain identical information for color transformations. In order to reduce printer memory requirements and PostScript transmission times it is advantageous to synchronize profiles and CRDs.

CRDs should be created with an optional `CreationDate` field indicating the date/time of CRD creation or most recent modification. This should correspond to the date/time field of any companion profiles. This profile information is available in the profile's header and the `calibrationDateTimeTag`. Even if no companion profile is constructed, this field should still be supplied in CRDs. Companion means embodying the same transformation, but in a different format -- profile vs. CRD.

`CreationDate` is a PostScript string whose format closely follows the format defined by the International Standard ASN.1 (Abstract Syntax Notation One, defined in CCITT X.208 or ISO/IEC 8824). This string is of the form:

( YYYYMMDDHHmmSSOHH ' mm ' )

where

YYYY is the year

MM is the month (01-12)

DD is the day (01-31)

HH is the hour (00-23)

mm are the minutes (00-59)

SS are the seconds (00-59)

O is the relation of local time to GMT where + indicates that local time is later than GMT,  
- indicates that local time is earlier than GMT, and z indicates that local time is GMT

HH' is the absolute value of the offset from GMT in hours

mm' is the absolute value of the offset from GMT in minutes

Trailing fields other than the year are optional. The default value for day and month is 1; all other numerical fields default to 0. If no GMT information is specified, the relationship of the specified time to GMT is considered unknown. Whether the time zone is known or not, the rest of the date should be specified in local time.

Profiles should be extended with the optional tag `crdInfoTag`. This tag contains the PostScript product name to which this profile corresponds and the names of the companion CRDs. See 6.5.2 for the format.

`CreationDate` and `crdInfoTag` can be synchronized in different ways depending on the availability of bidirectional communications between the host and printer, as well as whether the CRD came with the printer or was downloaded from the host in the field.

When bidirectional communications are available one can query the printer and determine the availability of a given CRD and its associated `CreationDate`. When such communications are not available, the list of printer resident CRDs and their `CreationDate` entries are available through the printer's PPD and the host profile registry.

PPDs currently contain the names of the CRDs that ship with a printer. The PPD format will be extended to also contain the `CreationDate` entry for each CRD. For those CRDs downloaded (or deleted) in the field, the registry should be updated accordingly. The existence and form of this registry may differ from platform to platform.

There are 3 cases to consider.

*1) CRDs and profiles made together.*

If one wants to know whether it is necessary to construct and download a CRD for a given profile one can:

- a) Optionally check to see if the profile corresponds to the printer by comparing the PostScript product name field in the `crdInfoTag` and the printer's product name. The product name for the printer is determined using the PostScript product operator or from the PPD. One may want to perform this step to limit the user's selection of profiles to only those appropriate for the printer.

- b) Based on the desired rendering intent, check to see if the printer has a CRD of the appropriate name as found in crdInfoTag.
- c) If b) is successful, compare the profile's date/time field to the CRD's CreationDate. If a match, then the profile doesn't need to be downloaded -- the companion CRD already exists and is used for the job. If not successful in a), b) or c) see 2) below.

*2) CRDs generated from ICC profiles and then downloaded.*

One can download a CRD for just a job in which case future synchronization is not an issue. Or one can make the CRD persistent in the printer. In this case one fills in the CRD CreationDate field, updates the registry, and updates the profile to have the appropriate crdInfoTag tag.

*3) CRD in printer for which no profiles exist.*

This is mainly a problem for CRDs that exist today without companion profiles. If one wants to make a companion profile and have synchronization with CRDs one can:

- a) Use the CRD's CreationDate field if available for the date/time field of the profile.
- b) Update the CRD in the printer, and the registry, with the CreationDate corresponding to the new profile's date/time field.

In both cases the profile's crdInfoTag should be filled in appropriately.

This works like 1) with the exception that the CRD updates may be volatile to power cycles of the printer. Such power cycles should result in the registry being updated.

## Annex D

### Profile Connection Space explanation

#### D.1 Introduction

This Appendix is intended to clarify certain issues of interpretation in the ICC Profile Format.

The goal of color management is to provide the capability of maintaining control over color rendering among various devices and media that may be interconnected through a computer system or network. To achieve this goal, the color characteristics of each device are determined and encapsulated in a *device profile*, which is a digital representation of the relation between device coordinates and a device-independent specification of color.

By *device coordinates* we mean the numerical quantities through which a computer system communicates with a color peripheral—such as the digital code values used to drive a monitor or printer, or the digital signals received from a scanner. These quantities are usually labeled *RGB* (or *CMYK*), but the labels identify the channels of the device rather than specific visual colors; the quantities are often encoded as unsigned 8-bit integers for each channel in the typical digital interface.

The *device-independent specification* is best given in a color space based on human visual experience. Thus, a device profile provides a means of translating (or transforming) color image data from device coordinates into a visual color space or vice versa.

Furthermore, if the various profiles available to a color-management system are referenced to the *same* visual color space, the system can translate data from one device's coordinates to another's—while maintaining color consistency—by (conceptually) passing through the intermediary of the visual color space; the latter, then, constitutes a standard interface for color communication, allowing profiles to be connected together in a meaningful sequence. A color space used in this way may be termed a *Profile Connection Space* (PCS). For example, the transformation of a color image from a scanner into monitor coordinates can be described as a transformation into the PCS (via the scanner's device profile) followed by a transformation out of the PCS (via the monitor's device profile). In practice, these successive transformations may be implemented in a variety of ways, and the image may never actually be represented in the PCS on disk or in computer memory. Thus, the PCS is to be regarded as a convenient reference for the definition of profiles—as an intermediate, or virtual, stage of the image processing—, in contrast to an *interchange* or *exchange color space*, which is an encoding for the storage and transmission of color images. The issues regarding the choice or design of a PCS are somewhat different from those related to an interchange space; this annex is concerned only with PCS issues.

A PCS consists of a coordinate system for color space and an interpretation of the data represented in that coordinate system. In fact, multiple coordinate systems can easily be supported in the same or different color-management systems, as long as they share a common interpretation, since it is usually a well-defined and relatively simple mathematical task to transform from one coordinate system to another. However, if the interpretation of the represented colors is different, there may be no satisfactory way of translating the data from one to another.

The purpose of this paper is to present an unambiguous interpretation for the PCS implicit in the ICC Profile Format. It is especially important in the heterogeneous environments currently found on desktop platforms and networks to establish this interpretation in an open, non-proprietary specification, so that different color-management systems can communicate with each other and exchange profiles within and across platforms and operating systems.

## D.2 Colorimetry and its interpretation

The issue of interpretation has received little attention in the recent past, because it has been widely believed that the choice of a suitable coordinate system—preferably one founded on *CIE colorimetry*, a system of measurement and quantification of visual color stimuli created and promoted by the *Commission Internationale de l'Éclairage*—would suffice to guarantee device independence. The notion was that colorimetric matching of the renderings on various media was the key to satisfactory color reproduction, and that interpretation was not needed. However, although colorimetry can be an essential element of a successful approach to color management, it is usually necessary to modify the colorimetric specification for renderings on different media.

Different media require different physical color stimuli, in certain cases, because they will be viewed in different environments—e.g., different surround conditions or illuminants; the observers, therefore, will experience different adaptive effects. In order to preserve the same *color appearance* in these different environments, the colorimetry must be corrected to compensate for the adaptation of the human visual system and for physical differences in the viewing environments, such as flare. Although color appearance is still an active research topic, the most common forms of adaptation are understood reasonably well, so that the required corrections in the colorimetry for different viewing conditions can be modeled with sufficient accuracy.

There are other reasons why the colorimetry may be altered for specific media. For instance, hard-copy media—even those intended for the same viewing conditions—differ considerably in their dynamic range and color gamut. A well-crafted rendering of an image on a specific medium will take advantage of the capabilities of that medium without creating objectionable artifacts imposed by its limitations. For instance, the tone reproduction of the image should provide sufficient contrast in the midtones without producing blocked-up shadows or washed-out highlights. The detailed shape of the tone curve will depend on the minimum and maximum densities ( $D_{min}$  and  $D_{max}$ ) attainable in the medium. Clearly, there is considerable art involved in shaping the tone-reproduction and color-reproduction characteristics of different media, and much of this art is based on subjective, aesthetic judgments. As a result, the substrate (paper, transparency material, etc.) and the colorants used in a medium will be exploited to impart a particular “personality” to the reproduction that is characteristic of the medium.

Furthermore, the desired behavior of a color-management system depends strongly on artistic intent. If the output medium is identical to the input medium—say, 35-mm slides—, the desired behavior is typically to create a duplicate of the original. But if the two media are different, it is not so obvious what the default behavior should be. In some cases, the intent may be to retain all or part of the personality of the original; in other cases, it may be more important to remove the personality of the original and replace it with a fresh rendering that has the full personality of the output medium. Sometimes the simulation of a third medium may be important—as when an image is displayed on a monitor to preview a rendering on a dye-diffusion printer, retaining (as well as possible) the personality of an original image scanned from a photographic print! It is essential to the success of color-management systems that a broad range of options be kept open. The interpretation of the PCS merely defines the particular default behavior that will be facilitated by the system without explicit intervention by the application or user. Alternative behaviors are not excluded by this choice; they simply will not be the default and will require more work.

With this context in mind, we present the following interpretation:

### **The PCS represents desired color appearances.**

Here, the term *desired* is used to indicate that the interpretation is oriented towards colors to be produced on an output medium. It also is used to imply that these colors are not restricted by the limitations of any particular output medium. It is helpful here to conceptualize a “reference reproduction medium”, with a large gamut and dynamic range, as the target medium for the desired colors. Consequently, it is the



responsibility of the output device profiles to clip or compress these colors into the gamut of the actual output media. And, of course, “desired” also implies the expression of artistic intent.

The term *color appearance* is used to imply that adaptive effects are taken into account. Associated with the reference reproduction medium is a “reference viewing environment”. More precisely, therefore, the PCS represents the “desired color appearances” in terms of *the CIE colorimetry of the colors to be rendered on the reference medium and viewed in the reference environment*. Output profiles for media that are viewed in different environments are responsible for modifying the colorimetry to account for the differences in the observer’s state of adaptation (and any substantial differences in flare light present in these environments), so that color appearance is preserved. Similarly, input profiles are responsible for modifying the colorimetry of the input media to account for adaptation and flare; they also have the responsibility to account for the artistic intent implicit in the word “desired”.

We define the *reference reproduction medium* as an idealized print, to be viewed in reflection, on a “paper” that is a perfect, non-selective diffuser (i.e.,  $D_{min} = 0$ ), with colorants having a large dynamic range and color gamut. We define the *reference viewing environment* to be the standard viewing booth (ANSI PH-2.30); in particular, it is characterized by a “normal” surround—i.e., where the illumination of the image is similar to the illumination of the rest of the environment—, and the adapting illuminant is specified to have the chromaticity of D50 (a particular daylight illuminant).

### D.3 Color measurements

The PCS, so interpreted, represents colors for a hypothetical reference medium; device profiles must relate these colors to those that can be measured on real media. For consistency of results, these measurements must be made in accordance with the principles of CIE colorimetry.

For one particular class of media (namely, those intended for the graphic-arts), the colorimetry should conform to graphic-arts standards for color measurement <sup>1)</sup> Here, the illuminant is specified to be D50, so that no corrections need to be applied for chromatic adaptation. The colorimetry standard is based on a theoretical D50 illuminant, as defined by the CIE in the form of a tabulated spectral distribution. However, the fluorescent D50 simulators found in typical professional viewing booths have rather different spectral distributions, and the color stimuli produced can be noticeably different. <sup>2)</sup> Often, better results can be obtained by basing the colorimetry on the actual, rather than the theoretical, illumination source; unfortunately, there is no standardized, practically realizable source.

For other, non-graphic-arts media, the illuminant may be different from D50. In general, for best results, the actual illumination spectrum should be used in the color measurements. And if the chromaticity of the illuminant is different from that of D50, corrections for chromatic adaptation will be needed and will be incorporated into the device-profile transforms. This aspect of the PCS interpretation provides flexibility to the color-management system. For example, it will be possible to transform data from a medium intended for tungsten illumination to a medium intended for cool-white-fluorescent: the input profile handles the adaptation from tungsten to D50, and the output profile handles the adaptation from D50 to cool-white.

Since substantial flare (perhaps 2–3%) may be present in an actual viewing environment, <sup>3)</sup> the colorimetry is defined in an ideal, flareless measurement environment; in this way, difficult telescopic color measure-

---

<sup>1)</sup> IT8.7/3, “Graphic technology—Input data for characterization of 4-color process printing”, draft standard of Subcommittee 4 (Color) of ANSI Committee IT8 (Digital Data Exchange Standards), 14 December 1992, Paragraph 4.2.

<sup>2)</sup> D. Walker, “The Effects of Illuminant Spectra on Desktop Color Reproduction”, in *Device-Independent Color Imaging*, R. Motta and H. Berberian, ed., *Proc. SPIE*, **1909**, 1993, pp. 236–246.

ments in the viewing environment can be avoided, and simple contact instruments and/or controlled laboratory conditions can be used instead. (Corrections should be applied to the data for any appreciable flare in the actual measurement environment and instruments.)

#### D.4 Colorimetry corrections and adjustments in Output Profiles

The implications of this interpretation should be emphasized: the creator of a profile is obliged to correct and adjust the PCS data for various effects. Since the PCS is interpreted with an output orientation, we will first examine the nature of these corrections and adjustments for output profiles. Then, in the next section, we will discuss the consequences for input profiles.

Let us look at a number of possible output paths:

#### D.5 Output to reflection print media

Included here are computer-driven printers, off-press proofing systems, offset presses, gravure printing, photographic prints, etc. These are generally intended for “normal” viewing environments; but corrections may be needed—e.g., for chromatic adaptation, if the illuminant’s chromaticity is other than that of D50.

In the simplest scenario, the user desires to reproduce colors colorimetrically (aside from adaptive corrections) so as to attain an appearance match. A distinction can be made between “ICC-absolute” and “media-relative” colorimetry in this context. *ICC-absolute* colorimetry coincides with the CIE system: color stimuli are referenced to a perfectly reflecting diffuser. All reflection print media have a reflectance less than 1.0 and cannot reproduce densities less than their particular  $D_{min}$ . In a cross-rendering task, the choice of ICC-absolute colorimetry leads to a close appearance match over most of the tonal range, but, if the  $D_{min}$  of the input medium is different from that of the output medium, the areas of the image that are left blank will be different. This circumstance has led to the use of *media-relative* colorimetry, in which the color stimuli are referenced to the paper (or other substrate). This choice leads to a cross-rendering style in which the output image may be lighter or darker overall than the input image, but the blank areas will coincide. Both capabilities must be supported, since there are users in both camps. However, the default chosen for ICC is media-relative colorimetry.

This can be made more precise: the default “colorimetric” transform will effectively apply a scaling operation in the CIE 1931  $XYZ$  color space:

$$X_a = (X_{mw} / X_i) X_r \quad (D.1)$$

$$Y_a = (Y_{mw} / Y_i) Y_r \quad (D.2)$$

$$Z_a = (Z_{mw} / Z_i) Z_r \quad (D.3)$$

where  $(XYZ)_r$  are the coordinates of a color in the PCS,  $(XYZ)_a$  are the coordinates of the corresponding color to be produced on the output medium,  $(XYZ)_i$  are the coordinates of the lightest neutral represented in the PCS (namely, one with the chromaticity of D50 and a luminance of 1.0), and  $(XYZ)_{mw}$  are the coordinates of the output paper (or other substrate) *adapted to the PCS illuminant* (D50). Thus, the lightest neutral in the PCS will be rendered as blank paper—regardless of the reflectance or color cast of the paper—; other neutrals and colors will be rescaled proportionately and will be rendered darker than the paper. Output on different reflection print media will then agree with the PCS and with each other in media-relative colorimetry and, therefore, in relative appearance.

<sup>3)</sup> R.W.G. Hunt, *The Reproduction of Colour*, Fourth Edition, Fountain Press, 1987, pp. 52–53.

In other cases, the preference may be for ICC-absolute colorimetry. This means that, within the limitations of the output medium, the CIE colorimetry of the output image should agree with values represented in the PCS. I.e.,  $X_a = X$ ,  $Y_a = Y$ , and  $Z_a = Z$ . One way of achieving this result is to apply a separate transformation to the PCS values, outside of the device profile (e.g., in application or system software):

$$X' = (X_i / X_{mw}) X \quad (D.4)$$

$$Y' = (Y_i / Y_{mw}) Y \quad (D.5)$$

$$Z' = (Z_i / Z_{mw}) Z \quad (D.6)$$

The relative values,  $X' Y' Z'$ , can then be processed through the default colorimetric transform (i.e., they are effectively substituted for  $(XYZ)_r$  in Equations 1–3) to achieve the desired result.

This capability depends on the availability to the color-management software of the colorimetry of the paper. The `mediaWhitePointTag` in the profile can be used for this purpose and should represent the adapted, ICC-absolute colorimetry of the lightest neutral that the device and/or medium can render (usually the blank substrate).

In either case, it may happen that the dynamic range and/or color gamut of the output medium is not sufficient to encompass all the colors encoded in the PCS. Some form of clipping will then occur—in the highlights, in the shadows, or in the most saturated colors. While an appearance match may be achieved over much of a color space, there will be a loss of detail in some regions. If this is objectionable, the operator should have an option for selecting a more explicit form of gamut compression to be applied to the colors as part of the output profile. ICC supports two styles of controlled gamut compression—“perceptual” and “saturation”—in addition to the “colorimetric” option, which clips abruptly at the gamut boundary. (An important case requiring explicit gamut compression is that of input from a transparency, where the dynamic range, even of the corrected colors, may exceed that of any reflection print medium.) Note that an explicit compression maps colors from the dynamic range and gamut of the reference medium to the range and gamut of the actual medium, so that only  $(XYZ)_{D50}$ —i.e., the lightest PCS neutral—will be rendered as blank paper, just as in the media-relative-colorimetric case. This time, however, the entire tone scale may be readjusted, to keep the shadows from blocking up and to maintain proper midtones, and some in-gamut colors may be adjusted to make room for out-of-gamut colors.

## D.6 Output to transparency media

This category might include overhead transparencies and large-format color-reversal media, as well as slide-production systems. Transparency materials are normally intended to be viewed by projection (using a tungsten lamp) in a dim or darkened room; in some cases, however, they are placed on a back-lit viewer for display, and in others they are used as a graphic-arts input medium, in which case they are examined on a light box or light table with the aid of a loupe. Accordingly, there are several possible viewing conditions for transparencies, requiring somewhat different corrections.

Typical color-reversal films have a much larger dynamic range than reflection media and higher midscale contrast. Their tone-reproduction characteristics have evolved empirically, but it may be plausible to explain them as partially compensating for dark-surround adaptation and the flare conditions typical in a projection room. The state of brightness adaptation in a projection room is also different from that in a reflection environment. To the extent that these explanations are valid, the colorimetry should be corrected for these effects. Furthermore, in some of these environments the visual system is partially adapted to a tungsten source, and chromatic corrections should be applied for the difference between tungsten and D50.

A “colorimetric” rendering, in this case, will actually produce an appearance match to the colors in the PCS, rather than a colorimetric match—i.e., the colors measured on the resulting transparency will differ from

those encoded in the PCS, but will appear the same *when the transparency is viewed in its intended environment* as the PCS colors would if rendered on the reference medium and viewed in reflection.

Note that the lightest neutral, (XYZ)D50, will be rendered at or near  $D_{min}$  of the transparency in the default (media-relative) colorimetric transform. An ICC-absolute colorimetric rendering can be generated in software, as described above for reflection-print media.

Explicit gamut compression can be provided as an option; it normally would not be needed for images input from photographic media, but it may be useful for input from computer graphics, since some of the highly saturated colors available on a computer color monitor fall outside the gamut of transparency media.

## D.7 Negative media

Here the target colors are those of a reflection print to be made from the negative. No adaptive corrections are required, unless the print is intended to be viewed under an illuminant other than D50. Explicit gamut compression is a useful option, and both media-relative and ICC-absolute colorimetric matches can be provided as in the case of direct-print media.

## D.8 Monitor display

The viewing conditions of a CRT monitor may require some corrections to the colorimetry, due to the effects of surround and flare. Also, if the monitor's white point is other than D50, chromatic adaptation must be accounted for. When corrections for these effects are applied, the colors in the display should match the appearance of those in the PCS and should provide accurate and useful feedback to the operator.

In most cases, the rendering should be "colorimetric" (possibly including adaptive corrections), in order to achieve this result. (As for reflection print media, this would be "media-relative" by default, but "ICC-absolute" colorimetry is also supported.) In other cases (video production, perhaps), it may be more important to the user to create a pleasing image on the monitor (without having out-of-gamut colors block up, for instance) than to preserve an appearance match to the PCS; for that purpose, explicit gamut compression would be a useful option.

In many scenarios, the monitor display is not the end product, but rather a tool for an operator to use in controlling the processing of images for other renderings. For this purpose, it will be possible to simulate on the monitor the colors that would be obtained on various other output media. The PCS colors are first transformed into the output-device coordinates, using any preferred style of gamut compression. Then they are transformed back to the PCS by using the (colorimetric) inverse output transform. (These two steps can be replaced by an equivalent "preview" transform.) Finally they are transformed (colorimetrically) into monitor coordinates for previewing. The result of compression to the output gamut should then be visible in the displayed image.

## D.9 Colorimetry corrections and adjustments in Input Profiles

The purpose of an input profile is to transform an image into the PCS—i.e., to specify the colors that are desired in the output. Since there are many possible intentions that a user might have for these colors, we cannot impose many restrictions on the nature of the transforms involved. Bearing in mind the capabilities of the output profiles, as just outlined, we can suggest the possibilities available to various classes of input profiles.

## D.10 Scanned reflection prints

Here the intended viewing environment may be identical to the reference, but, if not, adaptive corrections should be applied to the colorimetry. In the simplest case, the profile may consist of a transformation from

scanner signals to the colorimetry of the medium. In this case, the personality of the input medium has been preserved. If the output rendering is also “colorimetric”, the result will be an appearance match to the original. Indeed, if the output medium is the same as the input medium, the result should be a close facsimile or duplicate of the original.

By default, the rendering is based on media-relative colorimetry, as discussed above. Therefore, it should be remembered, when creating an input profile, that the (XYZ)D50 point of the PCS will be mapped to the  $D_{min}$  of the output medium. This implies that the  $D_{min}$  of the input medium must be mapped to the (XYZ)D50 point of the PCS, in order to facilitate the duplication of an original and a media-relative colorimetry match when cross-rendering.

In order to enable the alternative of ICC-absolute colorimetry, the `mediaWhitePointTag` of the input profile should be used to specify the colorimetry of the paper. This allows the ICC-absolute colorimetry of the original to be computed from media-relative colorimetry represented in the PCS, by analogy to Equations 1–3 above. These ICC-absolute color stimuli can then be converted to media-relative colorimetry for output by using the “white point” field of the output profile in Equations 4–6.

There are other possibilities, however. The input profile could be designed to remove some or all of the personality of the input medium, so that the PCS encoding makes use of more of the gamut and dynamic range of the reference medium. In these cases, it will probably be best to choose some form of explicit gamut compression in the output profile. The result may differ in appearance considerably from the original and will constitute a fresh rendering tuned to the capabilities and limitations of the output medium.

In any case, a calibrated color monitor, if available, can be used to display an accurate preview of the result.

### D.11 Scanned transparencies

Since transparencies are intended for viewing in a variety of environments, different kinds of adaptive corrections may be applied to the colorimetry of the input medium to obtain colors in the PCS. For instance, the device profile might transform scanner signals into the colorimetry of a reference print that would have the same appearance in the reference environment as the transparency produces in a projection environment. (Note that there may be no actual reflection print medium that has sufficient dynamic range to reproduce all of these color appearances). In this scenario, the personality of the color-reversal film or other transparency material is retained, even though the colorimetry has been modified for the PCS; still, this may be loosely termed a “colorimetric” transform, since the only corrections are for flare and adaptation.

As in the case of input prints, there are other possibilities: some or all of the personality of the input medium can be removed, according to artistic intent, yielding different results, which also depend on the style of gamut compression selected for output.

Normally, the  $D_{min}$  of the input medium should be mapped to (XYZ)D50 in the PCS. The absolute, adapted XYZ of the  $D_{min}$  color is recorded in the “medium white point” tag.

### D.12 Scanned negatives

Photographic negatives, of course, are not intended for direct viewing. Therefore, the colorimetry that is relevant here might be that of a hypothetical reflection print made from the negative and intended for viewing in the reference environment. No adaptive corrections should be applied. The personality of the result is that of the negative-positive system as a whole. Again, other possibilities exist, depending on artistic intent.

### D.13 Computer graphics

Such imagery is usually synthesized in the *RGB* space of a display monitor that provides visual feedback to the operator. Thus, adaptive corrections may need to be applied to the colorimetry of the monitor to define the colorimetry of a reference print having the same appearance.

The personality here is that of the synthetic image on the monitor screen.

### D.14 Scene capture

This pathway refers to video cameras, electronic still cameras, and other technologies (such as Photo CD™) that provide a capability of approximately determining the colorimetry of objects in a real-world scene. In most cases, the tone scale must be adjusted to provide enough contrast for viewing the reference medium in the reference environment; the colorfulness of the image should also be enhanced somewhat for that environment. The personality of the result, of course, depends on the nature of these adjustments.

### D.15 Colorimetric input

In some cases, input colors are specified that are intended to be processed colorimetrically, without any tone shaping or chromatic enhancement. This might be the case, for instance, when a scene-capture device is used to record the colorimetry of real-world objects for scientific reasons, rather than for creating a pleasing reproduction. It may also be the case when particular spot colors are specified in colorimetric terms. In these cases, the specified colorimetric values are left intact in the transformation to the PCS; no adaptive corrections or adjustments are applied. The PCS values should be represented in media-relative colorimetry, and the “white point” tag specifies the reference point for the scaling. In some cases this reference point will have a luminance of 1.0, and there will be no difference between media-relative and ICC-absolute colorimetry. In other cases the reference point will have the colorimetry of (say) the paper stock used in a spot-color sample book or of a particular light neutral in a scene. In most of these cases, the preferred output rendering will also be “colorimetric”. By default, as before, this will entail media-relative colorimetry; ICC-absolute colorimetry can be achieved, outside of the default transforms, by taking account of the “white point” tags of the input and output profiles and converting appropriately.

An image of this kind can be said to have no personality.

As can be inferred from some of these examples, the user may have a choice of input profiles having different intents, as well as a choice among output transforms having different intents. The end result depends on both of these choices, which, for the most predictable color reproduction, should be made in coordination. To aid in this coordination, there are profile tags that specify the rendering intent and that distinguish between input transforms that are colorimetric (aside from possible corrections for flare and adaptation) and those that have applied adjustments to the colorimetry.

## D.16 Techniques for colorimetry corrections

As we have seen, if the viewing conditions of the medium are different from the reference (e.g., projected slides or video viewed in dim or dark surround), corrections to the colorimetry of the reproduction should be applied.<sup>1)</sup> These should be designed to correct for differences in the flare light present in these environments, as well as the effects of non-normal surround, brightness adaptation to the absolute radiant flux of the illumination, and any other effects that are found to be significant. And if the medium was intended to be viewed under an illuminant of different chromaticity than that of D50, the profile should incorporate corrections for chromatic adaptation; these can simply be based on a linear scaling in XYZ (which happens automatically in the CIELAB system); alternatively, it can be based on the linear Von Kries transformation<sup>2)</sup> (or, if preferred, a more sophisticated, nonlinear model of color appearance, such as that of Hunt<sup>3)</sup> or Nayatani<sup>4)</sup>).

If the creators of device profiles universally apply these corrections to their colorimetric data, the PCS will have a universal, unambiguous interpretation, and images rendered “colorimetrically” will evoke (as nearly as possible) the same appearance, regardless of the medium and viewing environment of the reproduction. In this way, the same image can be rendered on photographic transparency material, various reflective print media, CRT’s, etc., and will, by and large, appear similar to the viewer. This goal cannot be achieved simply by matching the colorimetry of the reproductions. Various forms of explicit gamut compression and input effects can be made available for situations where other goals are important; the recommended PCS interpretation does not limit these possibilities in any way: it merely facilitates the default behavior of the color-management system.

---

<sup>1)</sup> Hunt, *op. cit.*, pp. 56–61.

<sup>2)</sup> R.W.G. Hunt, *Measuring Color*, Ellis Horwood, pp. 70–71.

<sup>3)</sup> *Ibid.*, pp. 146–173.

<sup>4)</sup> Y. Nayatani, K. Takahama, and H. Sobagaki, “Prediction of color appearance under various adapting conditions”, *Color Res. Appl.*, **11**, 62 (1986).

## Annex E

### Chromatic Adaptation Tag

This document is intended to describe the Chromatic Adaptation Tag in more detail. The first part provides examples of popular adaptation methods used today in creating profiles and the steps to generate the proper matrix for them. The second part outlines the steps CMMs should take in dealing with the Chromatic Adaptation Tag

#### E.1 Calculating the Chromatic Adaptation Matrix

Although this is an optional tag, profile applications should include this tag to improve interoperability. If there is no chromatic adaptation required, because the actual viewing illuminant is D50, an identity matrix should be stored in this tag.

Examples of chromatic adaptation methods which can be described by the above matrix are the von Kries transformation, the linearized Bradford transformation (the same as linearized CIECAM97 transformation), and a linear re-scaling of CIE XYZ values, a.k.a “wrong” von Kries transformation.

##### E.1.1 von Kries transformation

The following von Kries matrix is used for computations of cone response values:

$$\begin{bmatrix} \rho \\ \gamma \\ \beta \end{bmatrix} = \begin{bmatrix} 0.40024 & 0.70760 & -0.08081 \\ -0.22630 & 1.16532 & 0.04570 \\ 0.0 & 0.0 & 0.91822 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = M_{vKries} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (\text{E.1})$$

The calculations of corresponding (visually equivalent) CIE XYZ values between two white points can be expressed in terms of the following matrix transformation:

$$\begin{bmatrix} X_{pcs} \\ Y_{pcs} \\ Z_{pcs} \end{bmatrix} = M_{vKries}^{-1} \begin{bmatrix} \rho_{pcs}/\rho_{src} & 0 & 0 \\ 0 & \gamma_{pcs}/\gamma_{src} & 0 \\ 0 & 0 & \beta_{pcs}/\beta_{src} \end{bmatrix} M_{vKries} \begin{bmatrix} X_{src} \\ Y_{src} \\ Z_{src} \end{bmatrix} \quad (\text{E.2})$$

Where:

$$\begin{bmatrix} \rho_{src} \\ \gamma_{src} \\ \beta_{src} \end{bmatrix} = M_{vKries} \begin{bmatrix} X_{WPsrc} \\ Y_{WPsrc} \\ Z_{WPsrc} \end{bmatrix} \quad (\text{E.3})$$

$$\begin{bmatrix} \rho_{pcs} \\ \gamma_{pcs} \\ \beta_{pcs} \end{bmatrix} = M_{vKries} \begin{bmatrix} X_{WPpcs} \\ Y_{WPpcs} \\ Z_{WPpcs} \end{bmatrix} \quad (\text{E.4})$$



$XYZ_{WPsrc}$  represents the illuminant of the actual device viewing condition.

Therefore, the chromatic adaptation matrix  $M_{adapt}$  can be derived by concatenating three matrices shown above, as follows:

$$M_{adapt} = M_{vKries}^{-1} \begin{bmatrix} \rho_{pcs}/\rho_{src} & 0 & 0 \\ 0 & \gamma_{pcs}/\gamma_{src} & 0 \\ 0 & 0 & \beta_{pcs}/\beta_{src} \end{bmatrix} M_{vKries} \quad (E.5)$$

### E.1.2 Linearized Bradford/CIECAM97 transformation

When full adaptation is assumed and a negligible non-linearity in the blue channel is omitted, the Bradford transformation is identical to the CIECAM97 transformation. Under the above assumption both become the same variant of a cone-space transform. Similarly as in von Kries method, the cone response values can be found through the matrix equation:

$$\begin{bmatrix} \rho \\ \gamma \\ \beta \end{bmatrix} = \begin{bmatrix} 0.8951 & 0.2664 & -0.1614 \\ -0.7502 & 1.7135 & 0.0367 \\ 0.0389 & -0.0685 & 1.0296 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = M_{BFD} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (E.6)$$

The calculations of corresponding (visually equivalent) CIE XYZ values between two white points are the same as in von Kries transformation and similarly, the following chromatic adaptation matrix can be derived:

$$M_{adapt} = M_{BFD}^{-1} \begin{bmatrix} \rho_{pcs}/\rho_{src} & 0 & 0 \\ 0 & \gamma_{pcs}/\gamma_{src} & 0 \\ 0 & 0 & \beta_{pcs}/\beta_{src} \end{bmatrix} M_{BFD} \quad (E.7)$$

Where:

$$\begin{bmatrix} \rho_{src} \\ \gamma_{src} \\ \beta_{src} \end{bmatrix} = M_{BFD} \begin{bmatrix} X_{WPsrc} \\ Y_{WPsrc} \\ Z_{WPsrc} \end{bmatrix} \quad (E.8)$$

$$\begin{bmatrix} \rho_{pcs} \\ \gamma_{pcs} \\ \beta_{pcs} \end{bmatrix} = M_{BFD} \begin{bmatrix} X_{WPpcs} \\ Y_{WPpcs} \\ Z_{WPpcs} \end{bmatrix} \quad (E.9)$$

$XY_{WPpcs}$  represents the illuminant of the reference viewing condition.

### E.1.3 “ Wrong” von Kries transformation.

This method consists of applying the von Kries type transformation to normalized CIE XYZ values instead of cone response tristimulus values. The calculation of corresponding (visually equivalent) CIE XYZ values between two white points are expressed in terms of the following matrix transformation:

$$\begin{bmatrix} X_{pcs} \\ Y_{pcs} \\ Z_{pcs} \end{bmatrix} = \begin{bmatrix} X_{WP_{pcs}}/X_{WP_{src}} & 0 & 0 \\ 0 & Y_{WP_{pcs}}/Y_{WP_{src}} & 0 \\ 0 & 0 & Z_{WP_{pcs}}/Z_{WP_{src}} \end{bmatrix} \begin{bmatrix} X_{src} \\ Y_{src} \\ Z_{src} \end{bmatrix} \quad (\text{E.10})$$

Where  $XYZ_{WP_{pcs}}$  represents the illuminant of the reference viewing condition and  $XYZ_{WP_{src}}$  represents the illuminant of the actual device viewing condition.

In this case the chromatic adaptation matrix is the above 3 by 3 matrix.

## E.2 Applying the Chromatic Adaptation Matrix

The CMM should look at the source and destination profiles to determine what adjustments can be made. There are several possibilities:

1. Neither profile has the chromaticAdaptationTag. No action can be taken.
2. Both profiles have the chromaticAdaptationTag. If the same method is used, no action should be taken. If different methods are used, undo them first before using a consistent method of the CMM's choice.
3. Only one profile has chromaticAdaptationTag. Processing is implementation dependent.

Here is a step by step example of how to do the adjustments in the CMM if the color transformation is created from two RGB Display profiles containing the chromaticAdaptationTag.

Step 1. Determine if the two methods are the same. If the two matrices are identical, the chromatic adaptation methods are the same. If the matrices are different, the methods could still be the same while the actual viewing illuminants are different. One easy way to test this is: if M1 and M2 represent the chromatic adaptation matrices from profile 1 and 2 respectively, it can be proven that chromatic adaptation algorithms are the same if the following matrix equation holds true:  $M1 * M2 == M2 * M1$ . We can stop here if two algorithms are the same.

Step 2. Determine the actual device viewing illuminant for profile 1. This can be achieved by applying the inverse chromatic adaptation matrix to the PCS D50 XYZ value.

Step 3. Invert the red, green, and blue values stored in the colorant tags to the actual device illuminant values. This is accomplished by applying the inverse of the chromatic adaptation matrix for each colorant.

Step 4. Calculate the new chromatic adaptation matrix. Follow the examples of E.1. Use your favorite cone response matrix and generate a new matrix.

Step 5. Generate new D50 relative colorant values for red, green, and blue by applying the matrix calculated in step 4 to colorant values in the device illuminant derived in step 3.

Step 6. Repeat steps 2 to 5 for profile 2.

For profiles with LUT tags, the adjustments can be made after the values are converted into the PCS by adding an extra processing step of undoing and redoing the chromatic adaptation.

## Annex F Summary of spec changes

This annex lists the major changes made to the specification for the past few spec revisions. Minor editorial and cosmetic changes are not listed.

These are the changes made from Revision ICC.1:1998-09

1. The descriptions of the various rendering intents have been clarified. (See 4.9 and all of A.3.)  
*Per: Online ballot #200015*
2. Two new attribute bits have been added to the profile header. (See Table 17.)  
*Per: Online ballot #199805*
3. The interpretation of multidimensional tags is now defined in more situations. (See Table 20 and all of Section 6.3)  
*Per: Online ballot #200016*
4. Multidimensional tags are now allowed in monochrome device profiles. (See 6.3.1.1, 6.3.2.1, and 6.3.3.1.)  
*Per: Online ballot #200016*
5. A new optional tag, chromaticAdaptationTag, has been added. (See 6.4.11 and Annex E.)  
*Per: Online ballot #200010*
6. A new optional tag, chromaticityTag, has been added. (See 6.4.12 and 6.5.1.)  
*Per: Online ballot #199908*
7. The description of gamutTag has been reworded to make clear that its input is PCS values. (See 6.4.18.)  
*Per: Spec Editing WG*
8. The descriptions of lut8Type and lut16Type have been expanded to explain how tables map to one another. (See 6.5.7, 6.5.8, and A.2.)  
*Per: Online ballot #199806*
9. The offset of the name prefix in namedColorType has been corrected to read "16..t". (See Table 56.)  
*Per: Spec Editing WG*
10. A rule was added to prevent the modification of certain text inside textDescriptionType strings. (See 6.5.17.)
11. The illumination level of the reference viewing condition is defined as 500 lux (previously it was undefined). (See A.1)  
*Per: Online Ballot #200101*
12. The term "relative colorimetry" has been changed to "media-relative colorimetry" and the term "absolute colorimetry" has been changed to "ICC-absolute colorimetry" to avoid confusion with CIE terminology. (See A.3.1.2 and A.3.1.3.)  
*Per: Online ballot #200015*
13. The C header file example has been removed from the specification. It can be found online at the ICC Web site.  
*Per: Spec Editing WG*
14. The formatting has been altered to more closely match the format of ISO and IEC standards. (Note: This specification is not an International Standard, and it does not meet all of the ISO/IEC drafting rules.)  
*Per: Spec Editing WG*

These are the changes made from Version 3.4 (August 1997):

1. The Normative References have been brought up to date and expanded to include all cited International Standards. (See clause 2.)  
*Per: Spec Editing WG*
2. The CMM Type and Primary Platform signatures are now allowed to be set to zero. (See 6.1.2 and 6.1.7.)  
*Per: Resolution voted 1998-03-15*
3. The profile version number in the header has been changed to 2.2.0. (See 6.1.3.)  
*Per: Resolution voted 1998-07-24*
4. The terms “low  $n$  bits” and “first  $n$  bits” have been changed to “least-significant  $n$  bits” to avoid confusion. (See 6.1.8, 6.1.10, 6.1.11, 6.2.2, and 6.5.)  
*Per: Spec Editing WG*
5. A tag can now only appear once in a profile. (See 6.2.)  
*Per: Resolution voted 1998-03-15*
6. The table describing the interpretation of context-dependent tags has been expanded to explicitly list the contexts where the interpretation is undefined. (See Table 20.)  
*Per: Resolution voted 1998-07-24*
7. The rules concerning which classes of profiles can and cannot be embedded in images have been made consistent. (See 6.3.4.2, 6.3.4.3 and B.)  
*Per: Resolution voted 1998-03-15*
8. A new optional tag, deviceSettingsTag, has been added. (See 6.4.17 and 6.5.6.)  
*Per: Online ballot #199706*
9. A new optional tag, outputResponseTag, has been added. (See 6.4.28 and 6.5.13.) The new basic numeric type response16Number has been added to support the responseCurveSet16Type. (See 5.3.2.)  
*Per: Online ballot #199801*
10. The possibility for alignment problems in crdInfoType has been pointed out. (See 6.5.2.)  
*Per: Spec Editing WG*
11. All curveType tags now must follow the same interpretation rules for zero-entry and one-entry tables. (See 6.5.3.)  
*Per: Resolution voted 1998-03-15*
12. The method for embedding profiles in GIF images has been added. (See B.5.)  
*Per: Online ballot #199704*
13. The C header file has been updated to reflect the addition of new tags.  
*Per: Spec Editing WG*

These are the changes made from Version 3.3 (November 1996):

1. The Definitions clause has been updated. (See clause 4.)  
*Per: Spec Editing WG*
2. A new clause has been added for symbols and abbreviations used in the Specification. Abbreviations that were previously under Definitions have been moved to the new clause. (See 5.2.)  
*Per: Spec Editing WG*
3. The dataType and textType descriptions now spell out how the data size is calculated. (See 6.5.4 and 6.5.18.)  
*Per: Spec Editing WG*
4. The method for embedding profiles in JFIF images has been added. (See B.4.)  
*Per: Online ballot #199701*

5. The C header file has been updated. The conditional compilation has been altered to provide a default definition of the data types in all circumstances. The typedef for `iccCrInfoType` has been altered, and comments have been added to explain its use.  
*Per: Spec Editing WG*

These are the changes made from Version 3.2 (November 1995):

1. The requirements for Input Profiles have changed. Instead of having categories for RGB and CMYK input devices, the requirements have been changed to cover “three-component matrix-based” profiles and “N-component LUT-based” profiles. (See 6.3.1.2 and 6.3.1.3.)  
*Per: Online ballot with results reported on 1996-07-15*
2. The list of color space signatures has expanded to include generic color spaces (2colorData to 15colorData). (See Table 13.)  
*Per: Resolution voted 1996-07-15*
3. The profile version number in the header has been changed to 2.1.0. (See 6.1.3.)  
*Per: Included in "NCLR" (generic color space) resolution*
4. A new optional tag, `crdInfoTag`, has been added. (See 6.4.14, 6.5.2, and C.1.)  
*Per: Resolution voted 1996-03-28*
5. The signature of `namedColor2Type` was incorrectly listed as ‘ncl’ in Version 3.2 of the spec. The correct signature is ‘ncl2’. (See Table 57.)  
*Per: Spec Editing WG*
6. The description of the PCS encodings has been rewritten to clarify some issues. The actual encodings have not changed. (See A.1.)  
*Per: Resolution voted 1996-07-15*
7. The possibility for alignment problems in `textDescriptionType` and `ucrbgType` has been pointed out. (See 6.5.17 and 6.5.20.)  
*Per: Resolution voted 1996-07-15*
8. The examples for embedding profiles in EPS files have been corrected to show the required colon (:) after `%%BeginSetColorSpace` and `%%BeginRenderingIntent`. (See B.2.)  
*Per: Spec Editing WG*