

Information technology — Font information interchange —

Part 3: Glyph shape representation

ICS 35.240.30

National foreword

This British Standard is the UK implementation of ISO/IEC 9541-3:1994+A2:2009. It supersedes BS ISO/IEC 9541-3:1994, which is withdrawn.

The start and finish of text introduced or altered by amendment is indicated in the text by tags **A1** **A1**. Tags indicating changes to ISO/IEC text carry the number of the ISO/IEC amendment. For example, text altered by ISO/IEC amendment 1 is indicated by **A1** **A1**.

The UK participation in its preparation was entrusted to Technical Committee IST/41, Document description and processing language.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee.

© BSI 2009

ISBN 978 0 580 61375 3

Amendments/corrigenda issued since publication

Amd. No.	Date	Comments
15966	29 September 2006	Implementation of ISO/IEC amendment 1:2005.
	31 August 2009	Implementation of ISO/IEC amendment 2:2009.

**Information technology — Font
information interchange —**

Part 3:
Glyph shape representation

*Technologies de l'information — Échange d'informations sur les
fontes —*

Partie 3: Représentation de la forme de glyphes



Reference number
ISO/IEC 9541-3:1994(E)

Contents

	Page
Foreword	viii
Section 1. General	
Introduction	1
1.1 Scope	1
1.2 Conformance	1
1.3 Normative references	1
1.4 Notation	1
1.5 Overview of glyph shape representation	2
1.6 Specification of glyph shapes (GSHAPES)	2
1.7 Extensions to the font interchange format	2
1.7.1 ASN.1	2
1.7.2 SGML	3
Section 2. Type 1 glyph shape representation	
2.1 Scope	4
2.2 Definitions	4
2.3 Overview of Type 1 glyph shape representation architecture	5
2.4 Type 1 glyph shape concepts	6
2.4.1 Glyph coordinate system	6
2.4.2 Glyph procedure language	6
2.4.3 Glyph procedure interpreter	6
2.4.4 Alignment position	6
2.4.5 Flat position	6
2.4.6 Overshoot position	6
2.4.7 Overshoot suppression	6
2.4.8 Alignment zone	7
2.4.9 Hints	7
2.4.10 Hint zone	9
2.4.11 Path direction	9
2.4.12 Reference point	9
2.4.13 Flex mechanism	9
2.4.14 Hint substitution	9
2.4.15 Bézier curve	9
2.5 Glyph procedure interpreter model	10
2.6 Type 1 shape properties (T1SHAPES)	11
2.6.1 General properties (T1GENERAL)	11
2.6.1.1 Password (PASSWORD)	11
2.6.1.2 Paint type (PAINTTYPE)	11
2.6.1.3 Unique identifier (UNIQUEID)	11
2.6.2 Typographic color properties (T1COLOR)	12
2.6.2.1 Top alignment zones (BLUEVALUES)	12
2.6.2.2 Bottom alignment zones (OTHERBLUES)	12
2.6.2.3 Family top alignment (FAMILYBLUES)	13
2.6.2.4 Family bottom alignment zones (FAMILYOTHERBLUES)	13
2.6.2.5 Overshoot suppression size (BLUESCALE)	14
2.6.2.6 Overshoot enforcement extent (BLUESHIFT)	15
2.6.2.7 Zone tolerance (BLUEFUZZ)	15
2.6.2.8 Stem width information (STEMWIDTHPROPS)	15
2.6.2.8.1 Standard horizontal width (STDHW)	16
2.6.2.8.2 Standard vertical width (STDVW)	16
2.6.2.8.3 Stem snap horizontal (STEMSNAPH)	16

	Page
2.6.2.8.4 Stem snap vertical (STEMSNAPV)	16
2.6.2.8.5 Force bold (FORCEBOLD)	17
2.6.2.8.6 Language group (LANGUAGEGROUP)	17
2.6.3 Glyph properties (T1GLYPH)	17
2.6.3.1 Glyph procedure encryption (GLYPHENCRIPT)	18
2.6.3.2 Glyph prefix length (LENIV)	18
2.6.3.3 Accent Component Table (ACCENTENCODING)	18
2.6.3.4 Subroutines (SUBRS)	18
2.6.3.5 Glyph procedure properties (GLYPHPROCPROPS)	18
2.6.3.5.1 Glyph name property (GNAME)	19
2.6.3.5.2 Glyph procedure (GLYPHPROC)	19
2.7 Glyph procedure semantics	19
2.7.1 Virtual machine	19
2.7.1.1 Glyph procedure tokens	20
2.7.1.2 Glyph procedure interpreter	20
2.7.1.3 Operand list	20
2.7.1.4 State variables	21
2.7.1.4.1 Current point	21
2.7.1.4.2 Escapement	21
2.7.1.4.3 Current path	21
2.7.1.4.4 Horizontal stems	21
2.7.1.4.5 Vertical stems	21
2.7.1.4.6 Dot section	21
2.7.1.5 Implicit results stack	21
2.7.2 Data types	21
2.7.2.1 Object types	21
2.7.2.1.1 Integer	21
2.7.2.1.2 Number	21
2.7.3 Glyph procedure operators	22
2.7.3.1 Operators for starting and finishing	22
2.7.3.1.1 Reference point and escapement (rpe)	23
2.7.3.1.2 Horizontal reference point and escapement (xrpe)	23
2.7.3.1.3 End glyph (endglyph)	24
2.7.3.1.4 Standard indexed accented glyph (siag)	24
2.7.3.2 Path construction operators	24
2.7.3.2.1 Closepath (closepath)	24
2.7.3.2.2 Horizontal lineto (hlineto)	24
2.7.3.2.3 Horizontal moveto (hmoveto)	25
2.7.3.2.4 Horizontal-vertical curveto (hvcurveto)	25
2.7.3.2.5 Relative lineto (rlineto)	25
2.7.3.2.6 Relative moveto (rmoveto)	25
2.7.3.2.7 Relative relative curveto (rrcurveto)	25
2.7.3.2.8 Vertical-horizontal curveto (vhcurveto)	25
2.7.3.2.9 Vertical lineto (vlineto)	25
2.7.3.2.10 Vertical moveto (vmoveto)	26
2.7.3.2.11 Set current point (setcurrentpoint)	26
2.7.3.3 Hint operators	26
2.7.3.3.1 Dot section (dotsection)	26

	Page
2.7.3.3.2 Horizontal stem (hstem)	26
2.7.3.3.3 Horizontal stem 3 (hstem3)	26
2.7.3.3.4 Vertical stem (vstem)	27
2.7.3.3.5 Vertical stem 3 (vstem3)	27
2.7.3.4 Arithmetic operator	27
2.7.3.4.1 Divide (div)	27
2.7.3.5 Subroutine and subroutine-related operators	27
2.7.3.5.1 Call subroutine (callsubr)	27
2.7.3.5.2 Return (return)	28
2.7.3.5.3 Call utility subroutine (callutlsubr)	28
2.7.3.5.4 Return value (retval)	28
2.8 Subroutines	28
2.8.1 Calling utility subroutines	29
2.8.1.1 Calling utility subroutine 0	29
2.8.1.2 Calling utility subroutine 1	29
2.8.1.3 Calling utility subroutine 2	29
2.8.1.4 Calling utility subroutine 3	29
2.8.2 Hint substitution (Utility subroutine 3)	30
2.8.3 Flex mechanism	30
2.8.4 First four Subrs entries	32
2.8.5 Composite glyphs	32
2.8.5.1 Using the siag operator to create composite glyphs	32
2.8.5.2 Using subroutines to create composite glyphs	32
2.9 Interchange format	33
2.9.1 Extensions to font interchange format for type 1 glyph shape information	33
2.9.1.1 ASN.1 interchange format	33
2.9.1.2 SGML interchange format	34
2.9.2 Interchange format for glyph procedures	36
2.9.2.1 Glyph procedure number representation	36
2.9.2.2 Glyph operator interchange representation	36
2.9.2.3 Glyph procedure encryption	36
Section 3. Type 2 glyph shape representation	
3.1 Scope	38
3.2 Definitions	38
3.2.1 bitmap	38
3.2.2 bounding box	38
3.2.3 pixel	38
3.3 Overview of Type 2 glyph shape representation architecture	38
3.4 Type 2 shape properties (T2SHAPES)	38
3.4.1 Pixel size (PXLSIZE)	39
3.4.2 Output device technology (ODTECH)	39
3.4.3 Glyph-shape properties	39
3.4.3.1 Glyph name (GNAME)	39
3.4.3.2 Bounding box offset (BBOFFSET)	39
3.4.3.3 Bounding Box (BBOX)	39
3.4.3.4 Bitmap (BITMAP)	40
3.5 Interchange format	40

	Page
Section 4: Open Type 3 glyph shape representation	41
4.1 Scope	41
4.2 Definitions	41
4.3 Overview of Open Type 3 glyph shape representation architecture	41
4.3.1 Difference from Type 1 glyph shape representation	41
4.3.2 Extended virtual machine	41
4.4 Open Type 3 glyph shape representation	41
4.4.1 Unchanged Type 1 glyph procedure operators	42
4.4.1.1 Unchanged operators for starting and finishing	42
4.4.1.1.1 Reference point and escapement (rpe)	42
4.4.1.1.2 Horizontal reference point and escapement (xrpe)	42
4.4.1.1.3 End glyph (endglyph)	42
4.4.1.1.4 Standard indexed accented glyph (siag)	42
4.4.1.2 Unchanged path construction operators	42
4.4.1.2.1 Closepath (closepath)	42
4.4.1.2.2 Horizontal moveto (hmoveto)	42
4.4.1.2.3 Relative moveto (rmoveto)	42
4.4.1.2.4 Vertical moveto (vmoveto)	42
4.4.1.2.5 Set current point (setcurrentpoint)	43
4.4.1.3 Unchanged arithmetic operator	43
4.4.1.3.1 Divide (div)	43
4.4.1.4 Unchanged subroutine and subroutine-related operators	43
4.4.1.4.1 Call subroutine (callsubr)	43
4.4.1.4.2 Return (return)	43
4.4.1.4.3 Call utility subroutine (callutilsubr)	43
4.4.2 Enhanced Type 1 glyph procedure operators	43
4.4.2.1 Enhanced path construction operators	43
4.4.2.1.1 Horizontal lineto (hlineto)	43
4.4.2.1.2 Horizontal-vertical curveto (hvcurveto)	44
4.4.2.1.3 Relative lineto (rlineto)	44
4.4.2.1.4 Relative-relative lineto (rrcurveto)	44
4.4.2.1.5 Vertical-horizontal curveto (vhcurveto)	44
4.4.2.1.6 Vertical lineto (vlineto)	45
4.4.2.2 Enhanced hint operators	45
4.4.2.2.1 Horizontal stem (hstem)	45
4.4.2.2.2 Vertical stem (vstem)	45
4.4.3 Additional glyph procedure operators	46
4.4.3.1 Additional path construction operators	46
4.4.3.1.1 Flex (flex)	46
4.4.3.1.2 Flex 1 (flex1)	46
4.4.3.1.3 Horizontal flex (hflex)	46
4.4.3.1.4 Horizontal Flex 1 (hflex1)	46
4.4.3.1.5 Horizontal-horizontal curveto (hhcurveto)	46
4.4.3.1.6 Relative curveto-lineto (rcurveline)	47
4.4.3.1.7 Vertical-vertical curveto (vvcurveto)	47
4.4.3.2 Additional hint operators	47
4.4.3.2.1 Hintmask (hintmask)	47
4.4.3.2.2 Counter mask (cntrmask)	47
4.4.3.2.3 Horizontal stem hintmask (hstemhm)	47
4.4.3.2.4 Vertical stem hintmask (vstemhm)	48
4.4.3.3 Additional arithmetic operators	48
4.4.3.3.1 Absolute (abs)	48

	Page
4.4.3.3.2 Add (add)	48
4.4.3.3.3 Drop (drop)	48
4.4.3.3.4 Duplicate (dup)	48
4.4.3.3.5 Exchange (exch)	48
4.4.3.3.6 Index (index)	48
4.4.3.3.7 Multiply (mul)	49
4.4.3.3.8 Negate (neg)	49
4.4.3.3.9 Random (random)	49
4.4.3.3.10 Roll (roll)	49
4.4.3.3.11 Square root (sqrt)	49
4.4.3.3.12 Subtract (sub)	49
4.4.3.4 Additional conditional operators	49
4.4.3.4.1 And (and)	49
4.4.3.4.2 Equal (eq)	50
4.4.3.4.3 If else (ifelse)	50
4.4.3.4.4 Not equal (not)	50
4.4.3.4.5 Or (or)	50
4.4.3.5 Additional storage operators	50
4.4.3.5.1 Get (get)	50
4.4.3.5.2 Put (put)	50
4.5 Deprecated operators in Open Type 3 glyph shape description	50
4.5.1 Dot section (dotsection)	51
4.5.2 Return value (retval)	51
4.5.3 Horizontal and vertical stem 3 operators	51
4.6 Interchange format	51
4.6.1 Extensions to font interchange format for Open Type 3 glyph shape information	51
4.6.1.1 ASN.1 interchange format	51
4.6.1.2 SGML interchange format	52
4.6.1.3 Interchange format for glyph procedures	52
Annex A (normative) Default Accent Component Table	53
Annex B (informative) Compatibility with the installed base	60
B.1 Compatibility with installed base of glyph procedure interpreters	60
B.1.1 GlyphEncrypt property (GLYPHENCRIPT)	60
B.1.2 lenIV property (LENIV)	60
B.1.3 Hint zone values for ghost stems	60
B.1.4 Round stem up (ROUNDSTEMUP)	60
B.1.5 MinFeature (MINFEATURE)	60
B.1.6 Password (PASSWORD)	60
B.1.7 Accent Component Table	61
B.1.8 Interchange format	61
B.1.8.1 ASN.1 interchange format	61
B.1.8.2 SGML interchange format	61
Annex C (informative) Composite glyphs	61
C.1 Composite glyphs	61
C.1.1 Using the siag operator to create composite glyphs	61
C.1.2 Using subroutines to create composite glyphs	63
Annex D (informative) Interchange sample	63
Alphabetical index	66
Figure 1 — Three alignment zones, showing flat and overshoot positions for top and bottom zones	7
Figure 2 — Hstem and vstem examples	7

	Page
Figure 3 — Glyph level hint zones: the relationship between stems, hint zones, and hint operator operands	8
Figure 4 — Overlap zones	8
Figure 5 — Hint zones and alignment positions	9
Figure 6 — Bézier curve with control points	10
Figure 7 — Type 1 glyph procedure interpreter model	10
Figure 8 — The virtual machine	20
Figure 9 — Glyph procedure interpreter state diagram	23
Figure 10 — Appropriate and inappropriate curves for the Flex mechanism	31
Figure C.1 — Composite glyph constructed with siag operator: component glyphs and final composite glyph	50
Figure C.2 — Construction of a composite glyph using subroutines	51
Figure D.1 — Simplified glyph for encoding example	52
Table 1 — Symbols preceding glyph procedure notation	22
Table 2 — Symbols following glyph procedure notation	22
Table 3 — Symbols preceding glyph procedure notation for Open Type 3 glyph shape representation	42
Table A.1	53

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 9541-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 18, *Document processing and related communication*.

ISO/IEC 9541 consists of the following parts, under the general title *Information technology — Font information interchange*:

- *Part 1: Architecture*;
- *Part 2: Interchange Format*;
- *Part 3: Glyph shape representation*;
- *Part 4: Application-specific requirements*.

Part 1 of ISO/IEC 9541 specifies the architecture of a font resource, i.e., the font description, font metrics, glyph description and glyph metrics properties required for font references and the interchange of font resources.

Part 2 of ISO/IEC 9541 specifies the interchange formats for font information, and the minimum subsets of that information required for interchange.

Part 3 of ISO/IEC 9541 specifies the architecture and interchange formats for glyph shape representations.

Part 4 of ISO/IEC 9541 specifies the architecture and interchange format extensions for application specific (e.g. typesetting of mathematics) requirements.

Annex A forms an integral part of this part of ISO/IEC 9541. Annex B, Annex C and Annex D are for information only.

Section 1. General

Introduction

The use of open networks for the interchange of documents in both office and publishing environments has shown the need for a mechanism enabling the interchange of font information.

It is foreseen that publishing and office technologies will merge and that this development will be facilitated by definition of a standard font resource architecture and a limited number of standard font resource interchange formats.

1.1 Scope

ISO/IEC 9541, as a whole, specifies the architecture of font resources, as well as the formats for font interchange among information processing systems. It also specifies the architecture and formats that can be used to construct font references in general electronic document interchange.

This part of ISO/IEC 9541 specifies the architecture and interchange formats of glyph shape representations.

Font resources represented using the architecture and interchange formats defined in parts 1 and 2 of ISO/IEC 9541 are used in various document processing environments in which ASN.1 or SGML parsing algorithms are recognized. The encoding of font resource information as defined in this part of ISO/IEC 9541 is specified in both ASN.1 and SGML representations for consistent generation of font resources for use in these processing environments.

1.2 Conformance

A font resource conforming to this part of ISO/IEC 9541 is a conforming ISO/IEC 9541 font resource. The font resource must conform to the conformance conditions stated in clause 2 of ISO/IEC 9541-2:1991. A conforming implementation of the glyph procedure interpreter shall have the following minimum capabilities:

- represent at least numbers in the range of $-8\,000$ to $+8\,000$ with at least 12 bits of fractional information;
- hold at least 24 objects in the operand list.

1.3 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 9541. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO/IEC 9541 are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/IEC 8824:1990, *Information technology — Open Systems Interconnection — Specification of Abstract Syntax Notation One (ASN.1)*.

ISO/IEC 8825:1990, *Information technology — Open Systems Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)*.

ISO 8879:1986, *Information processing — Text and office systems — Standard Generalized Markup Language (SGML)*.

ISO/IEC 9070:1991, *Information technology — SGML support facilities — Registration procedures for public text owner identifiers*.

ISO/IEC 9541-1:1991, *Information technology — Font information interchange — Part 1: Architecture*.

ISO/IEC 9541-2:1991, *Information technology — Font information interchange — Part 2: Interchange Format*.

ISO/IEC 10036:1993, *Information technology — Font information interchange — Procedure for registration of glyph and glyph collection identifiers*.

1.4 Notation

The formal structure of glyph shape properties is specified using the BNF notation described in clause 4 of ISO/IEC 9541.

1.5 Overview of glyph shape representation

Each glyph shape representation technique makes use of different properties in specifying glyph shapes and therefore has its own architecture and interchange format. In this part of ISO/IEC 9541 each glyph shape representation technique is defined in a separate section. The glyph shape representation currently defined is

— ISO Standard Type 1 Glyph Shape Representation (specified in section 2)

NOTE 1 This part of ISO/IEC 9541 may be extended in the future by the addition of further sections specifying additional glyph shape representation techniques.

Glyph shape representations are divided into two broad categories: outline and bitmap representations of glyph shapes.

An outline representation describes a glyph using a mathematical description of the edges of glyph shapes. This has the advantage of allowing transformations such as scaling, rotation, and skewing, and permits many variations of style without additional storage requirements. An outline format also facilitates incorporation of added scaling information, called hints, which aid in the preservation of proportions for all sizes of raster grids (however, their usefulness is not confined to raster devices). Hints can also aid in achieving nonlinear scaling as an optical correction for different absolute sizes of presented glyphs.

For raster devices, outline fonts are converted, after adjustments for scaling requirements, to bitmap representations for final imaging and presentation. However, the presentation of outline glyph shape descriptions is not limited to raster devices; it may also include vector devices such as plotters, signage cutters, engraving machines, or variable spot size raster and gravure devices. Different shape representation techniques may vary in their appropriateness for different presentation devices.

Bitmap representations describe the pattern of pels which are required for printing on raster devices. Bitmap glyph representations are less capable of being scaled or transformed in arbitrary ways while retaining a high standard of typographic quality. Bitmaps of glyph shapes can be represented either as ordered columns or rows of dots, or by a variety of schemes designed to provide more compact representations, particularly for larger sizes.

1.6 Specification of glyph shapes (GSHAPES)

Any font resource conforming to ISO/IEC 9541 and containing glyph shape information shall contain a GSHAPES property. GSHAPES is a property-list of shape-property-lists defining the sets of shape information associated with this font resource.

shapes-property-list := shapes-name, shapes-value-property-list

shapes-name := STRUCTURED-NAME -- ISO/IEC 9541-3//GSHAPES

shapes-value-property-list := (t1-shape-property-list | property-list)*

This architecture allows any glyph shape representation to be defined. The architecture for ISO/IEC 9541 Standard Shape Representation Type 1 is defined in section 2.

1.7 Extensions to the font interchange format

ISO/IEC 9541 font information shall be interchanged using either the ASN.1 or SGML forms defined in ISO/IEC 9541-2. These interchange formats include “markers” to include a definition for interchange formats for glyph shape information. These formats are defined in this clause with further definitions of the detailed format for each glyph shape representation included in equivalent clauses in each of the following sections.

1.7.1 ASN.1

ISO9541-GSHAPES { 1 0 9541 3 0 } DEFINITIONS := BEGIN

IMPORTS T1-Shape-Property-List FROM ISO9541-GST1 { 1 0 9541 3 0 0 }

Glyph-Shapes := SET {

t1-shape-property-list [0] EXTERNAL T1-Shape-Property-List OPTIONAL,
-- see Section 2 of this part

non-iso-properties [99] IMPLICIT Property-List OPTIONAL }

1.7.2 SGML

A₂ <!-- (c) International Organization for Standardization 2004 Permission to copy in any form is granted for use with conforming WebSGML systems and applications as defined in ISO 8879:1986(WWW), provided this notice is included in all copies. -->
<!-- Public document type definition. Typical invocation:
<!DOCTYPE gshapes PUBLIC "ISO 9541-3:1994 AM1:2004 and AM2:2008//DTD Glyph Shapes//EN">
-->

<!-- GLYPHSHAPES -->
<!ELEMENT gshapes (t1shapes | t2shapes | ot3shapes | niprop)+ >

<!-- Type 1 shape information. Typical invocation:
<!DOCTYPE t1shapes PUBLIC "ISO 9541-3:1994//DTD Type 1 Glyph Shapes//EN">
-->

<!-- Type 2 shape information. Typical invocation:
<!DOCTYPE t2shapes PUBLIC "ISO 9541-3:1994 AM1:2004//DTD Type 2 Glyph Shapes//EN">
-->

<!-- Open Type 3 shape information. Typical invocation:
<!DOCTYPE ot3shapes PUBLIC "ISO 9541-3:1994 AM2:2008//DTD Open Type 3 Glyph Shapes//EN">
--> **A₂**

Section 2. Type 1 glyph shape representation

2.1 Scope

This section specifies the architecture and interchange format of one standard Glyph Shape Representation: ISO/IEC 9541 Standard TYPE 1. This representation technique is appropriate for, but not limited to, presentation on raster devices of low, moderate, and high resolution.

2.2 Definitions

The following definitions are specific to this section.

2.2.1

ciphertext

information which has been encrypted

2.2.2

current point

the last point referenced by a path-construction operator in the glyph description language; this point may be either the end point of the most recently drawn line or curve or the point most recently positioned by an `rmoveto` or `setcurrentpoint` operator

2.2.3

encryption key

integer number required for encrypting or decrypting a glyph procedure

2.2.4

font program

a computer program incorporating a structured set of glyph procedures and descriptive data

2.2.5

ghost stem

an imaginary horizontal stem, applied only at the Max-Y and Min-Y extents of a glyph, for which an `hstem` hint operator must be specified if correct vertical alignment is required

2.2.6

glyph procedure

a computer program written using the standard glyph shape description operators defined in this section and represented in the interchange format defined herein

2.2.7

glyph procedure interpreter

a computer process capable of interpreting a glyph procedure for the purpose of constructing an outline representation of the glyph and the associated data structure required for scan conversion

2.2.8

hint

a procedural or declarative specification of information additional to the geometric shape of a glyph, which aids in the preservation of proportions and features of that glyph during rasterizing for presentation on a raster device

2.2.9

overshoot

the part of a rounded or pointed glyph extremity which extends slightly beyond the position of flat-shaped extremities; used to achieve optically correct vertical alignment of glyphs for the horizontal writing mode

2.2.10

path

a possibly disjoint set of subpaths and regions, constructed by one or more path construction operators; the set of all subpaths in a glyph is a single path

2.2.11

plaintext

information which is not encrypted

2.2.12

rasterization

a two-step process which involves determining the areas of a glyph which must be filled, and then creating a bitmap to represent those areas

2.2.13

region

a geometric area

2.2.14

segment

the curve or straight line generated by a single graphics drawing command; the curve or straight line between consecutive points on the path defining a glyph shape

2.2.15

snap

a mechanism for forcing a collection of glyph stems of varying nominal widths (in their device independent outline representation) to convert to the same width when converted to a bitmap representation

2.2.16

subpath

a sequence of connected straight or curved line segments constructed by one or more path construction operators

2.2.17

typographic color

the relative boldness of the presented font; based on the relative width of the dominant type of stems in the font; also a comparative measure of the surface area of the presented glyph shapes relative to the area of the presentation surface

2.3 Overview of Type 1 glyph shape representation architecture

Type 1 glyph shape representation is an outline font representation which uses graphical drawing operators to describe the shape of a glyph. This approach has the advantage that the glyph can be scaled, rotated, and transformed in various ways, and can be rasterized to create a bitmap for presentation. Properties defining glyph shapes may be either procedural constructs or additional declarative data defined at the font or glyph level.

The glyph procedures may contain optional declarative hints. Hints consist of information in the font program which assists the glyph procedure interpreter in preserving the proportions and features of a glyph as it is scaled and optimized for the pel grid of a raster device, but their usefulness is not limited to this application. Font-level hints may be defined for the entire font and help to control the alignment and stem widths. Glyph-level hints apply only to the glyph in which they are defined, and primarily help to control the rasterization of vertical and horizontal stems.

Glyph shapes are defined by procedural constructs and additional declarative data called *properties*. These additional data help to accomplish a variety of goals:

- General Properties help to identify the font program and the type of glyph shape representation used (see **2.6.1**).
- Typographic color properties help to control the apparent weight of scaled and presented glyphs as well as to achieve uniform stroke widths (see **2.6.2**).
- Glyph procedure properties primarily provide parameters and subroutines for use by glyph procedures (see **2.6.3**).

NOTE 2 Glyph shape representations which conform to this part of ISO 9541 constitute computer programs or procedures and as such may be subject to protection under laws covering intellectual property rights.

This section defines the properties of glyph shapes in the following clauses:

- 2.4** explains Type 1 glyph shape concepts
- 2.5** explains the glyph procedure interpreter model
- 2.6** defines Type 1 glyph shape properties

- 2.7 defines glyph-level procedure semantics
- 2.8 defines the use of subroutine procedures
- 2.9 defines the interchange format

2.4 Type 1 glyph shape concepts

The following clauses explain concepts related to the Type 1 glyph shape technology.

2.4.1 Glyph coordinate system

The Type 1 glyph shape technology uses a continuous glyph coordinate system as described in subclause 8.2 of ISO/IEC 9541-1:1991.

NOTE 3 Throughout this section, any example value specified as a constant number of units is based on an assumption of a RELUNITS value of 1 000.

2.4.2 Glyph procedure language

The glyph procedure language is used to specify glyph shapes and hinting parameters. This language allows the specification of one path, which may consist of multiple disjoint subpaths. After constructing the entire path, the glyph procedure interpreter causes the glyph to be filled or stroked as one entity, depending on the value of the PAINTTYPE property.

2.4.3 Glyph procedure interpreter

The glyph procedure interpreter is a virtual machine (see 2.7.1) which interprets glyph procedures written in the glyph procedure language.

2.4.4 Alignment position

An alignment position is a single position offset which refers to a y-coordinate at which glyph extremities may align in the y direction. Alignment positions are only used in pairs to define alignment zones (see 2.4.8) for font resources with WRMODENAME values of LEFT-TO-RIGHT or RIGHT-TO-LEFT. A font resource with an ALIGNNAME value of BASE-ALIGN has its baseline on the x-axis. Alignment positions are specified by the BLUEVALUES and OTHERBLUES font-level properties and are specified in pairs consisting of a *flat* and an *overshoot* position (see 2.4.5 and 2.4.6).

Font resources with a WRMODENAME value of TOP-TO-BOTTOM, or an ALIGNNAME value of CENTRE-ALIGN, such as those containing East Asian ideographic glyphs, generally have glyphs centered on a single alignment position (which need not be specified in the font), and do not have separate alignment positions for extremities of the glyph shapes.

2.4.5 Flat position

A flat position is the alignment position to which primarily flat shaped glyph extremities align. The judgment as to what a flat shape is depends on an artistic decision by the typeface designer. Extremities which are judged to be almost flat may also be considered to be appropriate to align to a flat position. The concept of a flat position is not applicable to fonts with an ALIGNNAME value of CENTRE-ALIGN, or a WRMODENAME value of TOP-TO-BOTTOM.

2.4.6 Overshoot position

The overshoot position is an alignment position associated with, and occurring just beyond, a flat position (see Figure 1), to which non-flat glyph extremities may align. The purpose is to allow glyphs of various shapes to appear to align precisely, allowing for optical illusions, in the vertical direction. The overshoot position is not applicable to fonts with an ALIGNNAME value of CENTRE-ALIGN, or a WRMODENAME value of TOP-TO-BOTTOM.

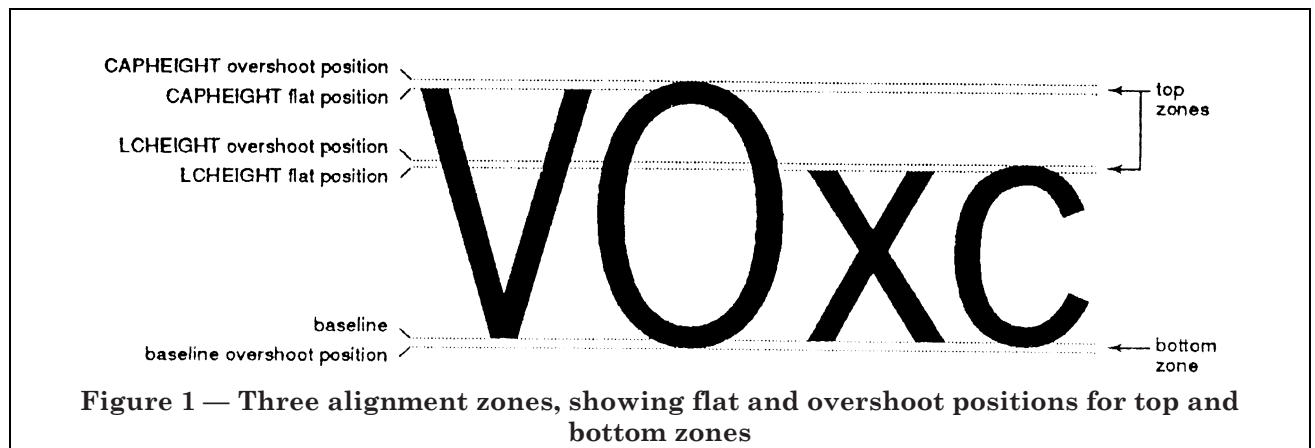
NOTE 4 Which glyphs require overshoots, and how much a glyph extremity extends past the flat position in the ideal outline representation, is a decision made by the typeface designer or developer. The Outline Modification and Intelligent Fill (rasterization) process (not part of this part of ISO/IEC 9541: see Figure 7) may control these extensions so that the resulting bitmap representation exhibits correct vertical alignment for the required size of glyph and characteristics of the presentation device.

2.4.7 Overshoot suppression

Overshoot suppression refers to a mechanism for not allowing overshoots to be rasterized at small sizes even though the scaling process might otherwise cause the overshoot features to round to one pel beyond the flat position.

2.4.8 Alignment zone

An alignment zone is the area lying between two alignment positions — beginning at a flat position and extending to the maximum overshoot position of the tops of glyphs (termed a *top zone*), or from a maximum overshoot position to the flat position of the bottoms of glyphs (a *bottom zone*). There may be multiple such zones. One bottom and two top zones are shown in Figure 1. The concept of an alignment zone is not applicable to fonts with an ALIGNNAME value of CENTRE-ALIGN, or a WRMODENAME value of TOP-TO-BOTTOM.



2.4.9 Hints

Hint information may be of two general types: font-level and glyph-level. Font-level hints are declared as properties and contain parameters which apply to all glyphs in the font. Font-level hints allow specification of alignment zones to control vertical alignment as well as to control stem widths for all glyphs in the font resource (see 2.6.2.8.1 through 2.6.2.8.5). Glyph-level hints are declared in glyph procedures, using glyph procedure operators, and apply only to the glyph procedure in which they are defined.

Within a glyph procedure, glyph-level declarative hints may be added to control the rasterization of horizontal and vertical stems. Stems may be either straight or curved. The hint operators indicate to the glyph procedure interpreter the *hint zone* (see 2.4.10) which extends from the coordinate specified by the first operand to the relative position indicated by the second operand. The *hstem* hint is used to control the number of pels, measured vertically, to which a horizontal stem is converted. The *vstem* hint is used to control the number of pels, measured horizontally, to which a vertical stem is converted. For the hint operators to be effective, the operands must refer to the exact coordinates of segment endpoints which are at the extremities of the stem.

This allows the glyph procedure interpreter to adjust the glyph outlines to help preserve design features and proportions for scaling to various sizes and resolutions. Figure 2 shows examples of *hstem* and *vstem* hints as applied to a Kanji glyph. This illustration shows a detail of the glyph which shows the path segment endpoints and the corresponding boundaries of the associated *hstem* hint zone.

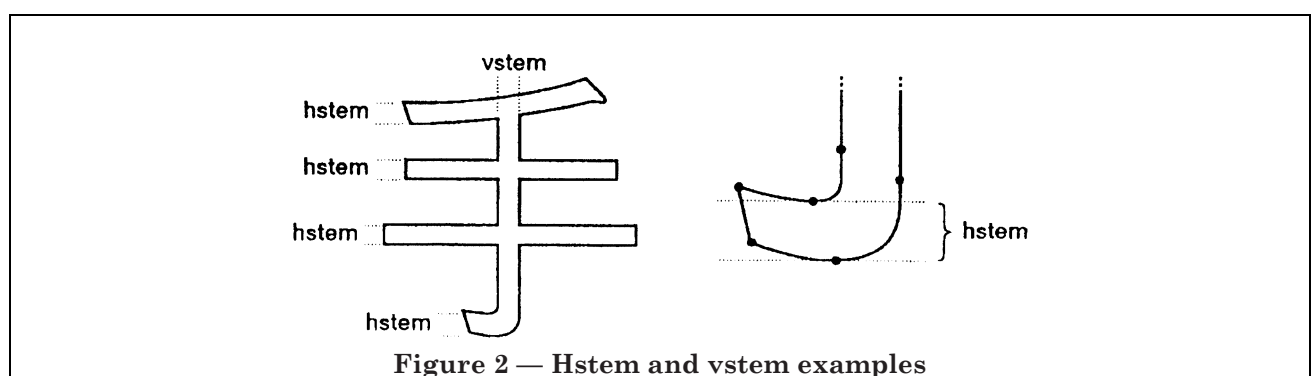
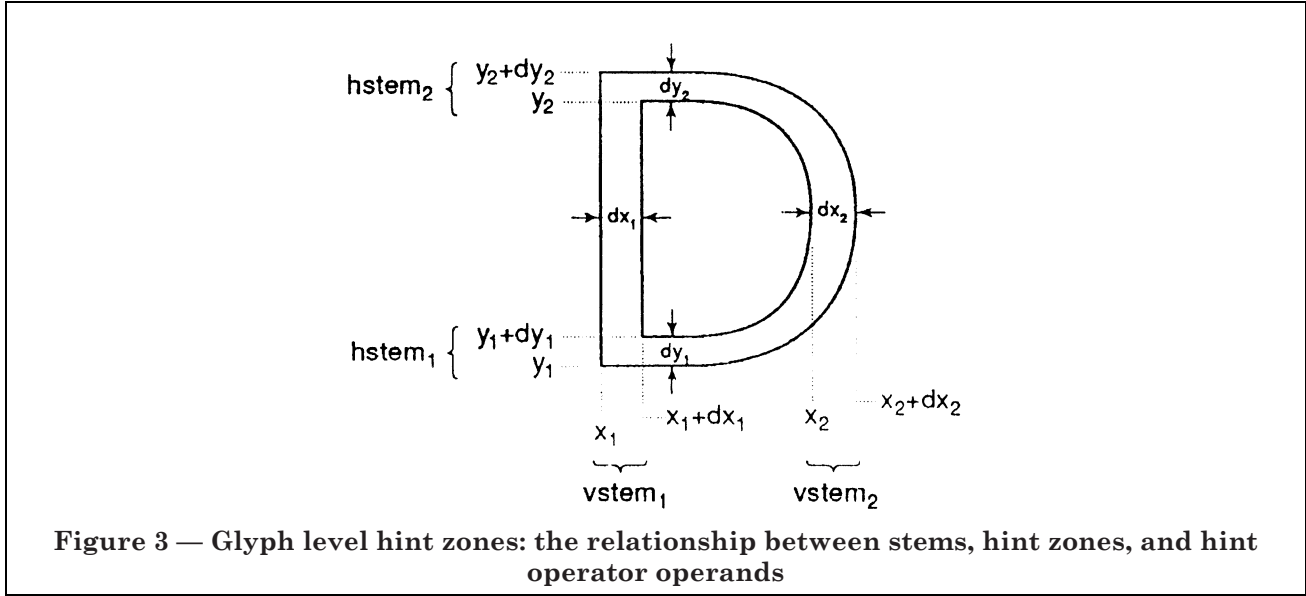
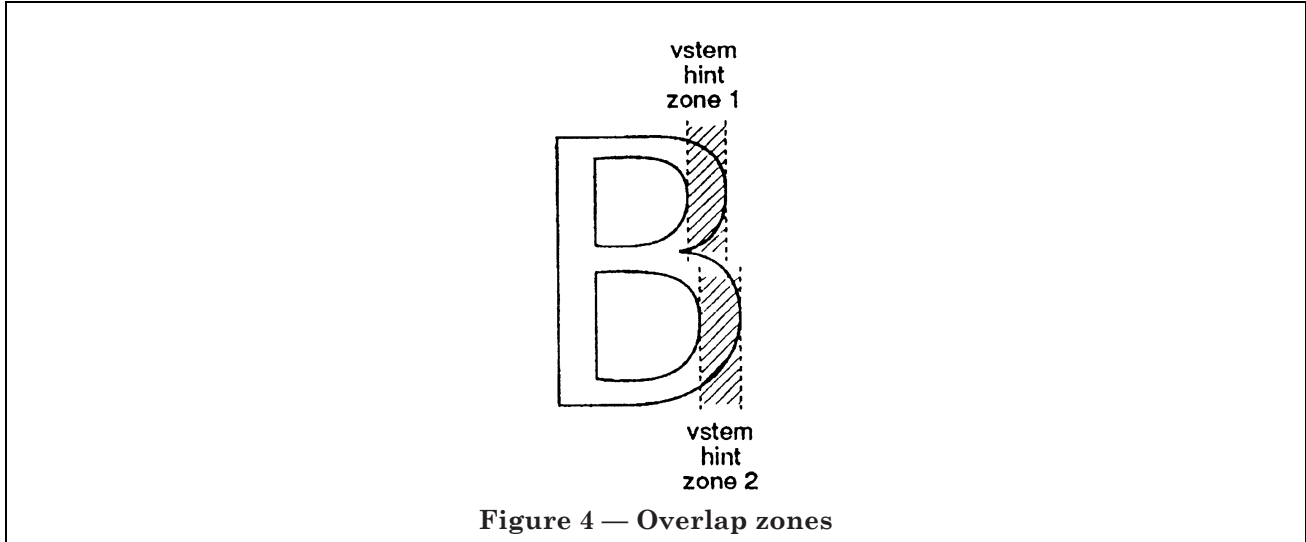


Figure 3 illustrates the appropriate hint zones for an uppercase “D” glyph, along with showing the hstem and vstem operand value relationships.



When hint zones with the same orientation (for example, two hint zones defined by two vstem hint operators) have overlapping coordinates within a glyph, as shown in Figure 4, proportions and features can best be preserved by replacing the originally declared hint zones with a new set at the appropriate place in the glyph drawing procedure. This process of hint substitution is described in 2.8.2.



A font which requires alignment to multiple alignment positions such as a Latin font (as opposed to a center-aligned script such as Kanji) shall have stem hints specified at, or in, a font-level alignment zone. In some glyphs this happens naturally. For example, Figure 5 shows an uppercase “E” where there are three hstem hints declared for the three horizontal stems. For a Latin font, the top stem’s hstem hint should extend to the capital height position, and the bottom stem’s hstem hint should extend down to the baseline.

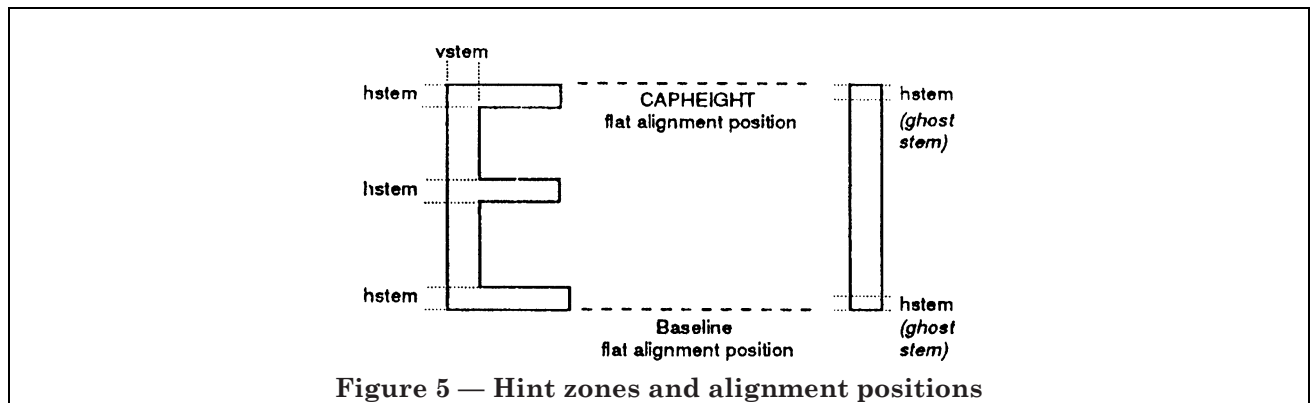


Figure 5 — Hint zones and alignment positions

In a sans serif uppercase “T”, however, there are no horizontal stems for these hstem hints. In order to have the capital height and baseline alignments apply to this glyph, it shall have hstem hints (called *ghost stems*, see the Definitions clause) defined for non-existent horizontal stems at these positions. If these hstem hints are not included in the glyph, the results are unspecified.

NOTE 5 For compatibility with the installed base of glyph procedure interpreters, the values allowed for the width of these ghost stems are restricted; see Annex B.

If a glyph is to align to font-level alignment positions, its hstem hint may not both have its top at or in a top zone and have its bottom at or in a bottom zone. In the uppercase “T” example, there cannot be just one hstem hint that stretches from baseline to the capital height line; instead one hstem should be at the capital height and another should be at the baseline.

NOTE 6 Several subclauses which discuss hint properties make recommendations for values for these properties. These are not mandatory values, but failure to follow the stated general guidelines may cause unexpected and possibly undesirable results on existing glyph procedure interpreters.

2.4.10 Hint zone

A hint zone is the area between and including the positions defined by a pair of operands to a glyph-level hint operator. The outer boundary specifies the exact location of path segment endpoints at the extremities of glyph features such as stems.

2.4.11 Path direction

A counterclockwise definition of a subpath indicates to the glyph procedure interpreter that the region is to be filled. A clockwise definition of a subpath indicates that the region is contained within a filled region and is to remain unfilled.

2.4.12 Reference point

The reference point is specified at the beginning of a glyph procedure by the operand of an rpe or xrpe operator. Each glyph procedure in a Type 1 font must specify a reference point; all coordinates specified in subsequent path construction or hint operators are interpreted as being relative to this point.

2.4.13 Flex mechanism

The Flex mechanism consists of a collection of three utility subroutines intended for use with shallow curves such as those used for cupped serifs. When these features are rasterized at small sizes, the Flex mechanism may be used to substitute a straight line. See 2.8.

2.4.14 Hint substitution

The hint substitution mechanism uses a utility subroutine to change the position of the hint zones for the current glyph. It is used when stems of the same orientation have overlapping coordinates within a single glyph. See 2.8.

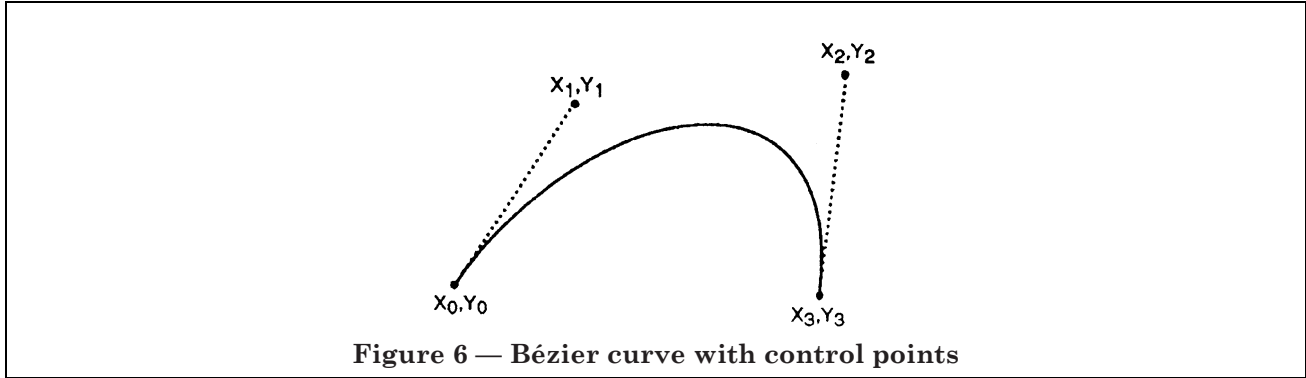
2.4.15 Bézier curve

A Bézier curve is derived from a pair of parametric cubic equations:

$$x(t) = a_x t^3 + b_x t^2 + c_x t + X_0 \quad \dots (2.1)$$

$$y(t) = a_y t^3 + b_y t^2 + c_y t + Y_0 \quad \dots (2.2)$$

The point X_0, Y_0 is the beginning point, and the point X_3, Y_3 is the end point. Associated with these are the control points $X_1, Y_1; X_2, Y_2$. The Bézier curve is tangent to the line segment from (X_0, Y_0) to (X_1, Y_1) at the point (X_0, Y_0) , and to the line segment from (X_2, Y_2) to (X_3, Y_3) at point (X_3, Y_3) . This curve will be referred to as the Bézier curve from (X_0, Y_0) to (X_3, Y_3) with control points (X_1, Y_1) and (X_2, Y_2) , as illustrated in Figure 6.

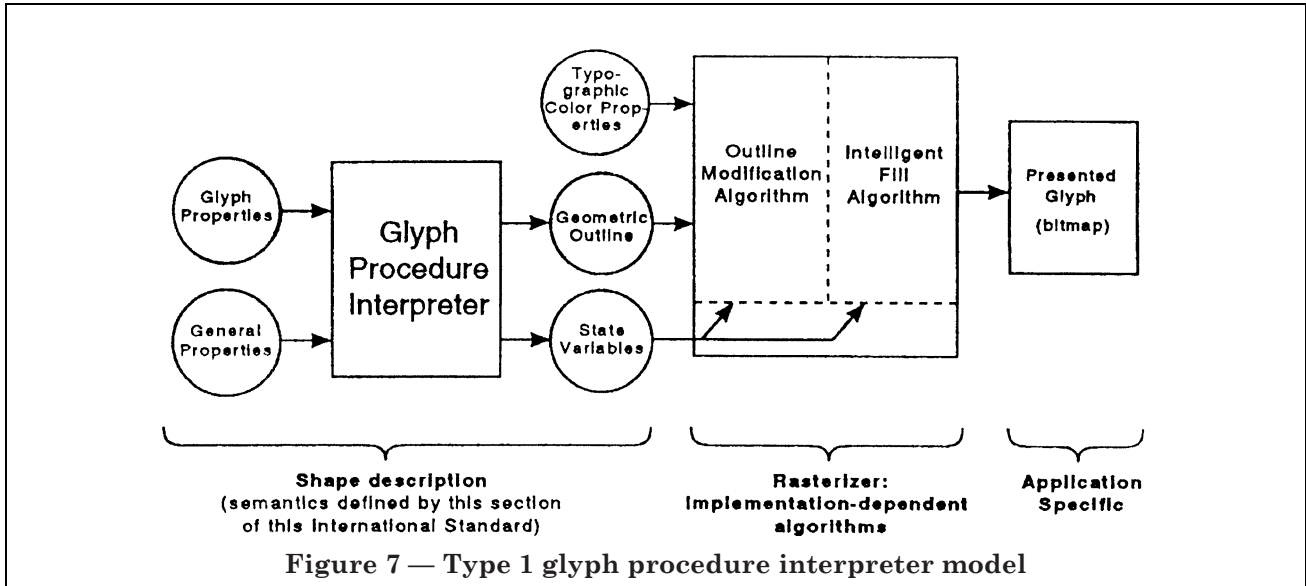


The Bézier control points corresponding to this curve are

$$\begin{aligned} X_1 &= X_0 + c_x/3 \\ Y_1 &= Y_0 + c_y/3 \\ X_2 &= X_1 + (c_x + b_x)/3 \\ Y_2 &= Y_1 + (c_y + b_y)/3 \\ X_3 &= X_0 + c_x + b_x + a_x \\ Y_3 &= Y_0 + c_y + b_y + a_y \end{aligned}$$

2.5 Glyph procedure interpreter model

Figure 7 shows a model of the glyph procedure interpreter as part of the process of preparing a glyph image for presentation. The glyph procedure interpreter interprets a glyph procedure and constructs the ideal glyph outline and a set of state variables. Only the semantics of this portion of the process are defined by this section. The Outline Modification Algorithm and Intelligent Fill Algorithm use the Geometric Outline, State Variables, and the Font-Level Hints as input to build the appropriate image for presentation. This part of the process is implementation-dependent, and is not defined by this part of ISO/IEC 9541.



2.6 Type 1 shape properties (T1SHAPES)

T1SHAPES is a property-list of shape properties, defining all the properties for Type 1 shape information. These consist of general font properties, typographic color properties, and glyph properties which have the following formal structure:

```

t1-shape-property-list ::= t1-shape-name,
                           t1-shape-value-property-list
t1-shape-name ::= STRUCTURED-NAME -- ISO/IEC 9541-3//T1SHAPES
t1-shape-value-property-list ::= ( t1-general-property-list,
                                   t1-color-property-list,
                                   t1-glyph-property-list )

```

2.6.1 General properties (T1GENERAL)

T1GENERAL is a list of global font properties which have the following formal structure:

```

t1-general-property-list ::= t1-general-name,
                             t1-general-value-property-list
t1-general-name ::= STRUCTURED-NAME -- ISO/IEC 9541-3//T1GENERAL
t1-general-value-property-list ::= ( password-property |
                                     painttype-property |
                                     uniqueid-property ) *

```

2.6.1.1 Password (PASSWORD)

The value of the optional PASSWORD property may be included to facilitate use of the font by existing glyph interpreters.

NOTE 7 The PASSWORD property only has significance with respect to compatibility with the installed base. Refer to Annex B for the value required for this property for compatibility purposes.

The formal structure of the PASSWORD property is

```

password-property ::= password-name, password-value
password-name ::= STRUCTURED-NAME -- ISO/IEC 9541-3//PASSWORD
password-value ::= INTEGER

```

2.6.1.2 Paint type (PAINTTYPE)

The value of the mandatory PAINTTYPE property indicates how the glyph contours in the font program are to be rasterized; the value shall be one of the following: 0 = filled; 2 = stroked.

NOTE 8 This property does not indicate how the glyphs will appear when presented, only how the path is treated by the scan conversion process: either filled or stroked. This property is different from the ISO/IEC 9541-1:1991 STRUCTURE property which specifies only whether the final appearance is a solid or outline glyph.

The formal structure of the PAINTTYPE property is

```

painttype-property ::= painttype-name, painttype-value
painttype-name ::= STRUCTURED-NAME -- ISO/IEC 9541-3//PAINTTYPE
painttype-value ::= INTEGER 0 | 2

```

2.6.1.3 Unique identifier (UNIQUEID)

The value of the optional UNIQUEID may be used to identify the font for the purpose of caching the bitmaps generated from glyph outlines.

NOTE 9 The primary purpose of the UNIQUEID property is to help to uniquely identify bitmaps created and cached from a font program. This is necessary because there is no guarantee that a font's name is unique. A UNIQUEID may allow bitmaps to be cached, either in printer memory or on a hard disk, for all subsequent print jobs instead of only for the current document.

The formal structure of the UNIQUEID property is

```

uniqueid-property ::= uniqueid-name, uniqueid-value
uniqueid-name ::= STRUCTURED-NAME -- ISO/IEC 9541-3//UNIQUEID
uniqueid-value ::= STRUCTURED-NAME

```

2.6.2 Typographic color properties (T1COLOR)

The typographic color properties may be conceptually divided into two categories: those that define font-level alignment zones and control overshoots, and those that help control stem widths of rasterized glyphs. The alignment and overshoot control properties are generally not applicable to fonts having a value of CENTRE-ALIGN for their ALIGNNAME property, or a WRMODENAME value of TOP-TO-BOTTOM. The BLUEVALUES property is the only mandatory typographic color property; it shall be included even if the list is empty. Optional properties with no values shall not be included. The stem control category of properties is applicable to any font resource.

The formal structure of the T1COLOR property-list is

```
t1-color-property-list      ::= t1-color-name,  
                             t1-color-value-property-list  
t1-color-name              ::= STRUCTURED-NAME  
                             -- ISO/IEC 9541-3//T1COLOR  
t1-color-value-property-list ::=  
    ( bluevalues-property | otherblues-property |  
      familyblues-property | familyotherblues-property |  
      bluescale-property | blueshift-property |  
      bluefuzz-property | stemwidthprops-property ) *
```

NOTE 10 The typographic color properties have names such as “BLUEVALUES” for historical and compatibility reasons. The term *Blue* was originally used in industry as part of a color-coding scheme for the digitization of typefaces. It has proved useful both to group these properties and to link them to the concept of helping to control the typographic color of a font.

2.6.2.1 Top alignment zones (BLUEVALUES)

The value of the mandatory BLUEVALUES property is a value list of pairs of integers which define alignment positions for one bottom zone (the baseline) and up to six top zones. Each integer in each pair of integers represents a vertical position in the glyph coordinate system. The sign of the values is positive in the positive y direction for font resources with WRMODENAME values of LEFT-TO-RIGHT and RIGHT-TO-LEFT; it is not defined for other values. The values shall obey the following rules:

- The first integer in each pair shall be less than or equal to the second integer in that pair.
- The first pair shall be the baseline overshoot and baseline positions.
- Different pairs shall be at least 3 glyph coordinate system units apart from each other and from pairs in the OTHERBLUES list. This minimum distance can be modified by the optional BLUEFUZZ property.
- The maximum difference between values in one pair is constrained as described under the description of the BLUESCALE property.

NOTE 11 Examples of top zone alignment positions which might be defined for Latin scripts may include: LCHEIGHT and LCHEIGHT overshoot position, ascender height and ascender height overshoot position, CAPHEIGHT and CAPHEIGHT overshoot position, figure height and figure height overshoot position, superior flat and superior overshoot position, and ordinal flat and ordinal overshoot position.

An empty list indicates that no alignment zones are necessary for the font. The BLUEVALUES property shall be declared even if the list is empty.

The formal structure of the BLUEVALUES property is

```
bluevalues-property ::= bluevalues-name, bluevalues-ordered-value-list  
bluevalues-name     ::= STRUCTURED-NAME -- ISO/IEC 9541-3//BLUEVALUES  
bluevalues-ordered-value-list ::= (INTEGER, INTEGER) *  
                                -- Maximum of 7 pairs
```

2.6.2.2 Bottom alignment zones (OTHERBLUES)

The value of the optional OTHERBLUES property is a value list of pairs of integers which define alignment positions for up to five bottom zones. Each integer in each pair of integers represents a vertical position in the glyph coordinate system. These bottom zones shall not duplicate the baseline zone required for the BLUEVALUES property. The sign of the values is positive in the positive y direction for font resources with WRMODENAME values of LEFT-TO-RIGHT and RIGHT-TO-LEFT; it is not defined for other values.

The values shall obey the following rules:

- The first integer in each pair shall be less than or equal to the second integer in that pair.
- Different pairs shall be at least 3 glyph coordinate system units apart from each other and from pairs in the BLUEVALUES list. This minimum distance can be modified by the optional BLUEFUZZ property.
- The maximum difference between values in one pair is constrained as described under the description of the BLUESCALE property.

NOTE 12 Typical examples of bottom zone alignment positions for Latin glyphs may include: descender depth overshoot position and descender depth, superior baseline overshoot position and superior baseline, and ordinal baseline overshoot position and ordinal baseline.

The formal structure of the OTHERBLUES property is

```
otherblues-property ::= otherblues-name, otherblues-ordered-value-list
otherblues-name ::= STRUCTURED-NAME -- ISO/IEC 9541-3//OTHERBLUES
otherblues-ordered-value-list ::= (INTEGER, INTEGER) *
                                -- Maximum of 5 pairs
```

2.6.2.3 Family top alignment (FAMILYBLUES)

The value of the optional FAMILYBLUES property is a list of pairs of integers which define alignment positions which shall be coordinated with those of the standard typeface in a family of typefaces. Each integer in each pair of integers represents a vertical position in the glyph coordinate system. This list specifies the reference values for one bottom zone (the baseline) and up to six top zones, and the contents of this list shall conform to the rules set forth in the list in 2.6.2.1.

The sign of the values is positive in the positive y direction for font resources with WRMODENAME values of LEFT-TO-RIGHT and RIGHT-TO-LEFT; it is not defined for other values.

For family alignment coordination to operate correctly, the following shall be observed with respect to the FAMILYBLUES property:

- One typeface of each family of typefaces is chosen as the standard for that family.
- If all faces in the family are to have their alignments coordinated, the value of the BLUEVALUES property of the standard face shall be copied from the standard face to be the value of the FAMILYBLUES property.

If the difference between an alignment position in a font and the corresponding family standard alignment position (as defined by FAMILYBLUES) is less than 1 pel, the family standard alignment position shall be used instead of the normal alignment position for that font.

NOTE 13 The FAMILYBLUES property is a means for coordinating the rasterized heights of glyphs from different styles of the same font family. When different styles of the same font family are mixed in text, it is often desirable to coordinate their alignment positions so they will be consistent at small sizes. If these entries are not present, then only a font program's own alignment hints will be considered. Family alignment positions are identical to individual font alignment positions; that is they are items such as the lowercase height and lowercase height overshoot.

The formal structure of the optional FAMILYBLUES property is

```
familyblues-property ::= familyblues-name, familyblues-ordered-value-list
familyblues-name ::= STRUCTURED-NAME -- ISO/IEC 9541-3//FAMILYBLUES
familyblues-ordered-value-list ::= (INTEGER, INTEGER) *
                                -- Maximum of 7 pairs
```

2.6.2.4 Family bottom alignment zones (FAMILYOTHERBLUES)

The value of the optional FAMILYOTHERBLUES property is a value list of pairs of integers which define alignment positions which shall be coordinated with those of the standard typeface in a family of typefaces. Each integer in each pair of integers represents a vertical position in the glyph coordinate system. This list specifies the reference values for up to five bottom zones, and the contents of this list shall conform to the rules set forth in the list in 2.6.2.2.

The sign of the values is positive in the positive y direction for font resources with WRMODENAME values of LEFT-TO-RIGHT and RIGHT-TO-LEFT; it is not defined for other values.

For family alignment coordination to operate correctly, the following shall be observed with respect to the FAMILYOTHERBLUES property:

- One typeface of each family of typefaces is chosen as the standard for that family.
- If all faces in the family are to have their bottom zone alignments coordinated, the value of the OTHERBLUES property of the standard face shall be copied from the standard face to be the value of the FAMILYOTHERBLUES property.

If the difference between an alignment position in a font and the corresponding family standard alignment position (as defined by FAMILYOTHERBLUES) is less than 1 pel, the family standard alignment shall be used instead of the normal alignment for that font.

NOTE 14 The FAMILYOTHERBLUES property is a means for coordinating the rasterized heights of glyphs from different styles of the same font family. When different styles of the same font family are mixed in text, it is often desirable to coordinate their alignment positions so they will be consistent at small sizes. If these entries are not present, then only a font program's own alignment hints will be considered. Family alignment positions are identical to individual font alignment positions; that is they are items such as descender depth, superior baseline, and ordinal baseline.

The formal structure of the FAMILYOTHERBLUES property is

```
familyotherblues-property := familyotherblues-name,  
                             familyotherblues-ordered-value-list  
familyotherblues-name := STRUCTURED-NAME  
                        -- ISO/IEC 9541-3//FAMILYOTHERBLUES  
familyotherblues-ordered-value-list := (INTEGER, INTEGER) *  
                                       -- Maximum of 5 pairs
```

2.6.2.5 Overshoot suppression size (BLUESCALE)

The value of the optional BLUESCALE property specifies the upper limit for the size at which glyph overshoot features shall be rasterized at the same position as the corresponding flat position. This property is not relevant for font resources having an ALIGNNAME property with a value of CENTRE-ALIGN, or for a WRMODENAME value of TOP-TO-BOTTOM.

- For glyph sizes such that RELUNITS glyph coordinate system units occupy fewer device pels than indicated by the BLUESCALE value, overshoot suppression is performed. All features falling in an alignment zone are rasterized at the same pel height.
- For glyph sizes such that RELUNITS glyph coordinate system units occupy the same or greater number of device pels than indicated by the BLUESCALE value, overshoot suppression is turned off, thus allowing overshoots to occur. (This behavior may be modified by the optional BLUESHIFT value; see 2.6.2.6.)

The BLUESCALE value is a number directly related to the number of vertical pels that RELUNITS glyph coordinate space units will occupy before overshoot suppression is turned off. A simple formula that relates this number of pels to the BLUESCALE value is

$$BLUESCALE = pels / RELUNITS \quad \dots (2.4)$$

NOTE 15 To turn off overshoot suppression at a nominal 46 pels per 1 000 glyph coordinate system units, BLUESCALE should be set to 46/1 000. With this setting of BLUESCALE, overshoot suppression will turn off at proportionately smaller sizes on higher resolution output devices or larger point sizes on lower resolution devices such as screen displays.

The product of the maximum alignment zone height and the pel size for overshoot suppression shall be less than RELUNITS.

NOTE 16 This restriction ensures that overshoot suppression will turn off before the overshoot alignment zone reaches a full device pel. For example, if a font contained a maximum alignment zone height of 23 and RELUNITS = 1 000, then the overshoot suppression size can be 43 pels but not 44.

The formal structure of the BLUESCALE property is

```
bluescale-property := bluescale-name, bluescale-value  
bluescale-name := STRUCTURED-NAME -- ISO/IEC 9541-3//BLUESCALE  
bluescale-value := REL-RATIONAL
```