VARC Hint Guidance for CFF2

Skef Iterum

February 12, 2024

CFSH — Compact Font Format Supplementary Hint Table

(This section is added as a new table specification.)

'CFSH' is an optional table used to provide supplementary 'CFF2'-format PrivateDICT structures for VARC composite glyphs, and to map each VARC glyph to those PrivateDICTs or to PrivateDICTs in the 'CFF2' table. Its contents are either 'CFF2' subtables or slight modifications of those tables.

CFSH Header

Type	Name	Description
uint16	major_version	Table major version number (=1)
uint16	minor_version	Table minor version number (=0)
Offset32	privateDICTIndexOffset	Offset (from start of CFSH table) to a 'CFF2' INDEX of Private DICTs. 0 if no PrivateDICTIndex.
uint16	initialPrivateDICT	The FontDICT index associated with of the first entry in the PrivateDICTIndex (default is 0).
Offset32	fdSelectOffset	Offset (from start of CFSH table) to the FontDICTSelect subtable. Must not be 0.
Offset32	itemVarStoreOffset	Offset (from start of CFSH table) to the Item Variation Store table (may be 0)

Private DICT Index and initialPrivateDICT

A CFF2 Font DICT INDEX contains FontDICT structures, each of which encodes the size and offset of a Private DICT as is described in (crossreference section on CFF2 Private DICTs). CFSH eliminates this indirection through the FontDICT. Instead the Private DICT Index is a CFF2 Index structure (cross-reference section on CFF2 Indexes) that stores a list of Private DICTs directly, each of which is identified by an unsigned integer.

The initialPrivateDICT field is the "FontDICT index" associated with the first entry in the Private DICT Index. This will normally be set to the size of the CFF2 FontDICT INDEX, so that the index of the first CFSH Private DICT is one greater than the index of the last CFF2 Private DICT. The indexes of the two sets of Private DICTs must not overlap.

If the itemVarStoreOffset field is non-zero, then the `vsindex` and `blend` operators refer to the Item Variation Store it points to. If the field is zero then those operators refer to the Item Variation Store in the CFF2 table.

PrivateDICTs in the CFSH table must not use the `Subrs` operator, and thus are always self-contained.

The FontDICTSelect Offset

The FontDICTSelect offset points to a CFF2 FontDICTSelect subtable (cross-reference 'CFF2' section on FDSelect). A client should ignore this field when it is set to 0 to support future minor extensions of the table. However, in a version 1.0 'CFSH' table the offset should not be 0. As of version 1.0 only FontDICTSelect format 4 as described in (cross-reference section on FDSelect format 4) is supported, but with one modification: it is not required that the `fd` field in the first Range4 record be 0.

The FontDICTSelect subtable in 'CFSH' can overlap with the FontDICTSelect subtable in the 'CFF2' table but there must not be gap between the last glyph mapped in 'CFF2' and the first glyph mapped in 'CFSH'. The `sentinel` field in 'CFSH' must be the highest GID defined in the font.

The itemVarStore Offset

When not zero this points to an Item Variation Store used for the `vsindex` and `blend` operators for the private dicts in the table.

VARC and Hinting

(This will be a new, appropriately placed section in the VARC chapter.)

When a VARC composite glyph is built from 'glyf' components that include TT instructions, or from 'CFF2' components that include hinting parameters, that data should be considered part of the composite. Whether and how hint data is used when rasterizing the glyph can depend on a number of factors including the transforms applied to the component, both within VARC or "externally" (e.g. using a CSS transform).

The 'glyf' table includes its own composite format that makes use of the TT instructions of component glyphs, and which can serve as a model for when and how to apply instructions when rasterizing a 'glyf'-based VARC composite.

Neither CFF nor CFF2, in contrast, has internal support for compositing, and while the 'COLR' table implicitly adds such support, that specification leaves the question of hinted output open. The rest of this section clarifies how to adapt 'CFF2' hinting parameters when rasterizing a VARC composite glyph.

Hinting CFF2 components in a VARC context

The hinting parameters in the 'CFF2' table evolved from related information in PostScript Type 1 fonts, and consists of a combination of "PrivateDict" parameters—which in CFF2 apply to subsets of glyphs—and per-glyph parameters. With VARC the PrivateDict parameters are taken from the PrivateDICT associated with the composite glyph; if there are multiple layers of compositing, the outer-most composite. The per-glyph parameters for a CFF2 component are adapted from those embedded in the CharString, relative to the cumulative transformations and translations through all compositing layers.

The Private DICT

Two subtables of two different tables define the association between the GID of a VARC composite and a Private DICT. The first is the FontDICTSelect subtable of the 'CFSH' table, which is checked first. If the GID is not mapped there, the mapping falls back to the FontDICTSelect subtable of 'CFF2'.

Note that rendering a CFF2 component of a composite glyph will typically require data from both the component's Private DICT and the composite's Private DICT. The former is always stored in the CFF2 table and mapped by the CFF2 FontDICTSelect subtable, and may contain `Subrs` and `vsindex` operators needed to desubroutinize and resolve any `blend`s. The composite PrivateDICT can be in either the CFF2 table or the CFSH table and defines the composite glyph's "Blue" parameters and standard stem sizes. (When a VARC composite loads data from the CFF2 component with the same GID, tne FontDICTSelect mapping in CFSH will typically need to overlap with that of the CFF2 table for that glyph.)

Fonts that conform to the specification will map all GIDs to at least one Private DICT, but if a mapping is missing the client should use FontDICT index 0 for a 'CFF2' glyph rendered directly. For a non-empty VARC composite glyph with a missing mapping the client should use the initialPrivateDICT value from 'CFSH' or 0 if that table is not present.

To find the PrivateDICT associated with the FontDICT index for a GID, the FontDICT INDEX in the 'CFF2' table should be checked first, followed by the Private DICT INDEX in the 'CFSH' table. If the Private DICT is empty or missing the 'CFF2' glyph or 'VARC' composite should be rendered using the default Private Dict values and without any alignment zones or expected stem sizes.

Per-glyph parameters

A hinted glyph has some combination of these parameters:

1. Horizontal and vertical stem regions (hstem(hm), vstem(hm))
2. Hintmasks
3. Counter hinting (cntrmask)

These are described in (cross-reference section(s) on per-glyph hinting parameters).

Prior to rasterizing, the stem positions and widths for each dimension of a CFF2-based VARC component must either be *transformed* or *cancelled* depending on the total (top-to-bottom) transformation applied to the component. Intuitively, the stems for both dimensions must be cancelled when there is rotation that is not a multiple of 180 degrees, and the stems in a given dimension must be cancelled when there is any skew in the opposite dimension. Otherwise the stems must be adjusted according to the scaling and translation in each dimension.

Whether the stems of a dimension must be cancelled or adjusted, and what the adjustment should be, can be determined by performing the cumulative transformation on the three points p_1 (100, 0), p_2 (0, 0) and p_3 (0, 100) and considering the result. If the transformed points are p_1', p_2', and p_3' respectively, adjustment of the horizontal stems proceeds as follows:

If the line from p_1' to p_2' is not close enough to horizontal for hinting purposes (which may vary by rasterizer implementation), hinting of horizontal stems is cancelled. Otherwise hinting proceeds in that dimension with a scale factor of

$s = (y_3' - y_2') / 100$
and a translation factor of
$t = y_2'$
If hinting proceeds, the horizontal stem deltas are unpacked into positional bottom, top pairs and each pair is processed as follows:

1. If the pair does not represent an edge stem, each value is multiplied by *s* and translated by *t*. If *s* is negative the top and bottom values are swapped.

2. If the pair represents a bottom edge stem, the lower edge is multiplied by *s* and translated by *t*. If *s* is positive the pair is re-encoded as a bottom edge stem with the adjusted value as the bottom. If *s* is negative the pair is re-encoded as a top edge stem with the adjusted value as the top.

   For example, if the original bottom position was *b*, the unpacked pair was encoded as having a width of -21, with the first number unpacking to $b + 21$ (the "upper edge") and the second to *b* (the "lower edge"). If *s* is positive the adjusted lower edge will be $b * s + t$ and the adjusted upper edge will be $b * s + t + 21$. If necessary this can be reencoded as a bottom edge hint with the first value $b * s + t + 21$ and a width of -21.

   If *s* is negative then the hint is adjusted to be a top hint, with $b * s + t$ as the upper edge and $b * s + t - 20$ as the lower edge, which can be re-encoded as an initial value of $b * s + t$ and a width of -20.

3. If the pair represents a top edge stem, the upper edge is multiplied by *s* and translated by *t*. If *s* is positive the pair is re-encoded as a top edge stem with the adjusted value as the top. If *s* is negative the pair is re-encoded as a bottom edge stem with the adjusted value as the bottom.

If *s* is positive the adjusted stem pairs simply replace the respective original pairs. If *s* is negative the orders of both the stem pairs and the corresponding bits in each of the hintmasks and cntrmasks for the glyph are reversed, and those stem pairs and masks are used to render the component.

Adjustment or cancellation of vertical stems is analogous to the horizontal case.

Adjustments to CFF2 chapter

(These aren't quite in "editorial form" yet, because the previous proposal that extensively modified the CFF2 text is not yet integrated.)

1. When a font has a VARC table, the length of the CFF2 CharStringINDEX can be less than the maxp count, otherwise it must be equal to the maxp count.

2. When a font has a VARC table, the highest glyph mapped by the FontDICTSelect structure can be less than the maxp count as long as two requirements are met. The first requirement is that there is a `CFSH` table that maps any remaining glyphs. The second requirement is that every glyph in the CharStringINDEX must be mapped.

The ranges of mapped glyphs can overlap as described in (cross-reference CFSH section “The FontDICTSelect Offset”)