The T1R encoding

Lars Hellström[*]

October 2, 2010

1 Specification

The T1R encoding is meant to be (almost) equivalent to the T1 encoding from the author's point of view but offer larger flexibility to font designers than the T1 encoding does—especially the flexibility to include additional ligatures. It is thus what I call a *relaxed* version of the T1 encoding, which is also the reason for its name.

Its basic derivation from T1 can be easily described: Slots 0–63, 64–127, and 192–255 have exactly the same contents as in the T1 encoding, whilst slots 128–191 can be used in any way the font designer wants; the set of commands, syntactic ligatures, and directly usable characters is exactly the same as in the T1 encoding. The result of my intention to give a formal description of the encoding might of course be that this description deviates slightly from the above, but it lines out the basic ideas.

The reasons for choosing this basis are pretty arbitrary, especially in terms of the assignment of slots. To begin with it gives a simple rule (the 256 slots in the font are divided into four continuous blocks of 64 slots each) and secondly almost all glyphs in slot ranges 0–63, 64–127, and 192–255 can be found in many popular fonts, but most of the glyphs in the slot range 128–191 are hard to find. Hence these glyphs will very frequently have to be made somehow anyway, the question is only at what time this will happen: At the time TEX is running, or afterwards, when a virtual font is interpreted? The character usually looks the same.

That is not the same thing as saying that it does not matter which, because it may well do. TEX will not automatically hyphenate words containing letters that are not a single character in the font, so some care is necessary. It is however the case that documents that both use and require hyphenation of all the accented characters that appear in slots 128–191 are extremely few (or do no exist at all—yet), so there are usually plenty of slots available for the font designer to put, for example, additional ligatures in. It might however well be that two different implementations of the same font (presumably made on different locations) have kept different sets of characters, due to the fact that the two implementors speak

[*] E-mail: Lars.Hellstrom@math.umu.se

different languages and hence need hyphentation of different sets of accented letters. This causes no problems (unless the implementors are exchanging `dvi` files), as the code in the font definition files would mirror this difference and set up the variable commands correctly anyhow, so that exchanging `.tex` manuscripts does not cause problems.

Now for the more formal description, with reservation for that (i) I might have missed some point about the `T1` encoding, lacking a formal definition of that and (ii) the current version of the `T1R` encoding is only a beta, so some details may well change in the future. In particular, the decisions about whether a composition should be a composition of a variant or a composition of the main variable command were usually pretty randomly made, so if someone should present me with a good reason why it should not be as it currently is, then there is a good chance I would change it^1.

1.1 The coding scheme

Every coding scheme for a `T1R` encoded font should comply to what is specified in Tables 1, 2, and 3. The corresponding table for slots 128–191 would read "not specified" in every entry in the glyph columns, so it has been omitted. 'not specified' means that the contents of that slot is not specified, so it is completely up to the font designer to decide, although there is always a default glyph even for the unassigned slots (viz. the same glyph as in the `T1` encoding). In most cases, there must be some font-dependent command variant definitions in the font definition file for each slot that deviates from the default.

The author is guaranteed to be able to access the characters in slots 32–126 simply through character tokens.

1.2 The syntactic ligatures

The required syntactic ligatures in the `T1R` encoding are listed in Table 4. Actually, the '`-`' (*hyphen*) + *hyphenchar* $\mapsto$ *hyphenchar* ligature is a kind of "semi-aestetic" ligature, as it might have an aestetic function as well as a syntactic. When TEX hyphenates a word, a *hyphenchar* is automatically inserted at the hyphenation point, but when the hyphenation point falls immediately after an explicit hyphen, this would result in two hyphens in the printed output, if it was not for this ligature. If you know what you do, you may change or even leave this ligature out, but be prepared that such an action must be attuned to, amongst other things, the hyphenation patterns and the current value of the TEX font parameter `\hyphenchar`.

Slot			Glyph	Slot			Glyph
Oct.	Dec.	Hex.		Oct.	Dec.	Hex.	
0	0	00	‘`’ (*grave accent*)	40	32	20	‘␣’ (*visible space*)
1	1	01	‘´’ (*acute accent*)	41	33	21	‘!’
2	2	02	‘ˆ’ (*circumflex accent*)	42	34	22	‘"’ (*quotedbl*)
3	3	03	‘˜’ (*tilde accent*)	43	35	23	‘#’
4	4	04	‘¨’ (*dieresis*)	44	36	24	‘$’
5	5	05	*Hungarian umlaut*	45	37	25	‘%’
6	6	06	‘˚’ (*ring accent*)	46	38	26	‘&’
7	7	07	‘ˇ’ (*caron accent*)	47	39	27	‘'’ (*quoteright*)
10	8	08	‘˘’ (*breve accent*)	50	40	28	‘(’
11	9	09	‘¯’ (*macron accent*)	51	41	29	‘)’
12	10	0A	*dot accent*	52	42	2A	‘*’ (*asterisk*)
13	11	0B	‘¸’ (*cedilla accent*)	53	43	2B	‘+’
14	12	0C	*ogonek accent*	54	44	2C	‘,’ (*comma*)
15	13	0D	*quotesinglbase*	55	45	2D	‘-’ (*hyphen*)
16	14	0E	*guilsinglleft*	56	46	2E	‘.’
17	15	0F	*guilsinglright*	57	47	2F	‘/’ (*slash*)
20	16	10	*quotedblleft*	60	48	30	‘0’ (*zero*)
21	17	11	*quotedblright*	61	49	31	‘1’
22	18	12	*quotedblbase*	62	50	32	‘2’
23	19	13	*guillemotleft*	63	51	33	‘3’
24	20	14	*guillemotright*	64	52	34	‘4’
25	21	15	*endash*	65	53	35	‘5’
26	22	16	*emdash*	66	54	36	‘6’
27	23	17	*compwordmark*	67	55	37	‘7’
30	24	18	*perthousandzero*	70	56	38	‘8’
31	25	19	‘ı’ (*dotlessi*)	71	57	39	‘9’
32	26	1A	‘ȷ’ (*dotlessj*)	72	58	3A	‘:’
33	27	1B	not specified	73	59	3B	‘;’
34	28	1C	not specified	74	60	3C	‘<’
35	29	1D	not specified	75	61	3D	‘=’
36	30	1E	not specified	76	62	3E	‘>’
37	31	1F	not specified	77	63	3F	‘?’

Table 1: The coding scheme for the T1R encoding, slots 0–63

Slot			Glyph	Slot			Glyph
Oct.	Dec.	Hex.		Oct.	Dec.	Hex.	
10	64	40	‘@’	140	96	60	‘`’ (*quoteleft*)
11	65	41	‘A’	141	97	61	‘a’
12	66	42	‘B’	142	98	62	‘b’
13	67	43	‘C’	143	99	63	‘c’
14	68	44	‘D’	144	100	64	‘d’
15	69	45	‘E’	145	101	65	‘e’
16	70	46	‘F’	146	102	66	‘f’
17	71	47	‘G’	147	103	67	‘g’
110	72	48	‘H’	150	104	68	‘h’
111	73	49	‘I’	151	105	69	‘i’
112	74	4A	‘J’	152	106	6A	‘j’
113	75	4B	‘K’	153	107	6B	‘k’
114	76	4C	‘L’	154	108	6C	‘l’
115	77	4D	‘M’	155	109	6D	‘m’
116	78	4E	‘N’	156	110	6E	‘n’
117	79	4F	‘O’	157	111	6F	‘o’
120	80	50	‘P’	160	112	70	‘p’
121	81	51	‘Q’	161	113	71	‘q’
122	82	52	‘R’	162	114	72	‘r’
123	83	53	‘S’	163	115	73	‘s’
124	84	54	‘T’	164	116	74	‘t’
125	85	55	‘U’	165	117	75	‘u’
126	86	56	‘V’	166	118	76	‘v’
127	87	57	‘W’	167	119	77	‘w’
130	88	58	‘X’	170	120	78	‘x’
131	89	59	‘Y’	171	121	79	‘y’
132	90	5A	‘Z’	172	122	7A	‘z’
133	91	5B	‘[’	173	123	7B	‘{’
134	92	5C	‘\’	174	124	7C	‘|’
135	93	5D	‘]’	175	125	7D	‘}’
136	94	5E	‘^’ (*circumflex character*)	176	126	7E	‘~’ (*tilde character*)
137	95	5F	‘_’	177	127	7F	*hyphenchar*

Table 2: The coding scheme for the T1R encoding, slots 64–127

Slot			Glyph	Slot			Glyph
Oct.	Dec.	Hex.		Oct.	Dec.	Hex.	
30	192	C0	‘À’	340	224	E0	‘à’
31	193	C1	‘Á’	341	225	E1	‘á’
32	194	C2	‘Â’	342	226	E2	‘â’
33	195	C3	‘Ã’	343	227	E3	‘ã’
34	196	C4	‘Ä’	344	228	E4	‘ä’
35	197	C5	‘Å’	345	229	E5	‘å’
36	198	C6	‘Æ’	346	230	E6	‘æ’
37	199	C7	‘Ç’	347	231	E7	‘ç’
310	200	C8	‘È’	350	232	E8	‘è’
311	201	C9	‘É’	351	233	E9	‘é’
312	202	CA	‘Ê’	352	234	EA	‘ê’
313	203	CB	‘Ë’	353	235	EB	‘ë’
314	204	CC	‘Ì’	354	236	EC	‘ì’
315	205	CD	‘Í’	355	237	ED	‘í’
316	206	CE	‘Î’	356	238	EE	‘î’
317	207	CF	‘Ï’	357	239	EF	‘ï’
320	208	D0	*Eth*	360	240	F0	*eth*
321	209	D1	‘Ñ’	361	241	F1	‘ñ’
322	210	D2	‘Ò’	362	242	F2	‘ò’
323	211	D3	‘Ó’	363	243	F3	‘ó’
324	212	D4	‘Ô’	364	244	F4	‘ô’
325	213	D5	‘Õ’	365	245	F5	‘õ’
326	214	D6	‘Ö’	366	246	F6	‘ö’
327	215	D7	‘Œ’ (*OE*)	367	247	F7	‘œ’
330	216	D8	‘Ø’	370	248	F8	‘ø’
331	217	D9	‘Ù’	371	249	F9	‘ù’
332	218	DA	‘Ú’	372	250	FA	‘ú’
333	219	DB	‘Û’	373	251	FB	‘û’
334	220	DC	‘Ü’	374	252	FC	‘ü’
335	221	DD	‘Ý’	375	253	FD	‘ý’
336	222	DE	*Thorn*	376	254	FE	*thorn*
337	223	DF	*SS*	377	255	FF	‘ß’ (*germandbls*)

Table 3: The coding scheme for the `T1R` encoding, slots 192–255

‘-’ (*hyphen*)	+	‘-’ (*hyphen*)	↦	*endash*
endash	+	‘-’ (*hyphen*)	↦	*emdash*
‘!’	+	‘‘’	↦	‘¡’
‘’’	+	‘’’	↦	*quotedblright*
‘,’	+	‘,’	↦	*quotedblbase*
‘-’ (*hyphen*)	+	*hyphenchar*	↦	*hyphenchar*
‘<’	+	‘<’	↦	*guillemotleft*
‘>’	+	‘>’	↦	*guillemotright*
‘?’	+	‘‘’	↦	‘¿’
‘‘’	+	‘‘’	↦	*quotedblleft*

Table 4: The required syntactic ligatures in the `T1R` encoding

No.	Meaning	No.	Meaning
1	Slant per pt	5	x-height (size of 1 ex)
2	Interword space	6	Quad width (size of 1 em)
3	Interword stretch	7	Extra space
4	Interword shrink		

Table 5: Required `\fontdimen`s in an `T1R` encoded font

1.3 The font dimensions

As for the required font dimensions, I have chosen to only require the seven font dimensions that all TEX fonts have in common. These are listed in Table 5. As both the `ec` family of fonts and the `T1` encoded fonts made by *fontinst* (as of v 1.8, see [1]) have seven more however, it seems like that set of fourteen font dimensions should be considered the current `T1` standard. I will probably have added the other seven by the time I get to the first non-beta release, but I have refrained from including them in this release since I do not feel sure enough about what they are to write a formal specification for them. It should be noted, though, that the specification only lists a minimal set of font dimensions—therefore including the other seven in a font is most likely only good.

1.4 The font-dependent commands

The font-dependent commands of the `T1R` encoding fall into two categories: symbol commands and accenting commands. The symbol commands simply typeset one symbol. The accenting commands take one argument and typesets the result of accenting the material that the argument would typeset.

A list of the symbol commands of the `T1R` encoding can be found in Table 6.

[1] Unless too many people's code have already come to depend on it. But in that case, there is always the possibility to make new encoding that differs from `T1R` only in a few such points and recommend people to use that instead.

Command	Action	Remark
\AE	Typeset 'Æ'	
\ae	Typeset 'æ'	
\DH	Typeset *Eth*	
\dh	Typeset *eth*	
\DJ	Typeset *Eth*	
\dj	Typeset *dbar*	Variable
\guillemotleft	Typeset *guillemotleft*	
\guillemotright	Typeset *guillemotright*	
\guilsinglleft	Typeset *guilsinglleft*	
\guilsinglright	Typeset *guilsinglright*	
\i	Typeset 'ı' (*dotlessi*)	
\j	Typeset 'ȷ' (*dotlessj*)	
\L	Typeset *Lslash*	Variable
\l	Typeset *lslash*	Variable
\NG	Typeset *Eng*	Variable
\ng	Typeset *eng*	Variable
\OE	Typeset 'Œ' (*OE*)	
\oe	Typeset 'œ'	
\O	Typeset 'Ø'	
\o	Typeset 'ø'	
\quotedblbase	Typeset *quotedblbase*	
\quotesinglbase	Typeset *quotesinglbase*	
\SS	Typeset *SS*	Variable
\ss	Typeset 'ß'	
\textasciicircum	Typeset '^' (*circumflex character*)	
\textasciitilde	Typeset '~' (*tilde character*)	
\textbackslash	Typeset '\'	
\textbar	Typeset '|'	
\textbraceleft	Typeset '{'	
\textbraceright	Typeset '}'	
\textcompwordmark	Typeset *compwordmark*	
\textdollar	Typeset '$'	
\textemdash	Typeset *emdash*	
\textendash	Typeset *endash*	
\textexclamdown	Typeset '¡'	Variable
\textgreater	Typeset '>'	
\textless	Typeset '<'	
\textperthousand	Typeset '%' followed by one *perthousandzero*	
\textpertenthousand	Typeset '%' followed by two *perthousandzero*	
\textquestiondown	Typeset '¿'	Variable
\textquotedblleft	Typeset *quotedblleft*	
\textquotedblright	Typeset *quotedblright*	
\textquotedbl	Typeset '"' (*quotedbl*)	
\textquoteleft	Typeset '‘'	
\textquoteright	Typeset '’'	
\textsection	Typeset *section*	Variable
\textsterling	Typeset *sterling*	Variable
\textunderscore	Typeset '_'	
\textvisiblespace	Typeset '␣'	
\TH	Typeset *Thorn*	
\th	Typeset *thorn*	

Table 6: The symbol commands of the T1R encoding

Command	Action	Remark
\`	Typeset the argument with a grave accent above it	Compositions for A, a, E, e, I, i, \i, O, o, U, and u
\'	Typeset the argument with an acute accent above it	Compositions for A, a, C (var.), c (var.), E, e, I, i, \i, L (var.), l (var.), N (var.), n (var.), O, o, R (var.), r (var.), S (var.), s (var.), U, u, Y, y, Z (var.), and z (var.)
\^	Typeset the argument with a circumflex accent above it	Compositions for A, a, E, e, I, i, \i, O, o, U, and u
\~	Typeset the argument with a tilde accent above it	Compositions for A, a, N, n, O, and o
\"	Typeset the argument with a dieresis accent above it	Compositions for A, a, E, e, I, i, \i, O, o, U, u, Y (var.), and y (var.)
\H	Typeset the argument with a Hungarian umlaut above it	Variable; compositions of the encoding level variant for O, o, U, and u
\r	Typeset the argument with a ring accent above it	Compositions for A, a, U (var.), and u (var.)
\v	Typeset the argument with a caron accent above it	Variable; compositions of the encoding level variant for C, c, D, d, E, e, L, l, N, n, R, r, S, s, T, t, Z, and z
\u	Typeset the argument with a breve accent above it	Variable; compositions of the encoding level variant for A, a, G, and g
\=	Typeset the argument with a macron accent above it	
\.	Typeset the argument with a dot accent above it	Variable; compositions of the encoding level variant for I, i, Z, and z
\b	Typeset the argument with a macron accent under it	
\c	Typeset the argument with a cedilla accent under it	Variable; compositions for C, c, S (var.), s (var.), T (var.), and t (var.)
\d	Typeset the argument with a dot accent under it	
\k	Typeset the argument with an ogonek accent under it	Variable; compositions of the encoding level variant for A, a, E, and e

Table 7: The accenting commands of the T1R encoding

One thing worth noticing about this table is that it lists some glyphs, for example *sterling*, that are not listed in the required coding scheme, yet the font designer is required to provide the author with these. Normally this would be done by simply including the glyph in the font in the same slot as in the T1 encoding, but if a large number of slots must be used for other glyphs, such as ligatures, then this may not be possible. In such cases the symbol command could instead typeset a symbol in *another* font. This could be done with a definition such as

```
\DefineTextCommandVariant{\textsterling}{T1R}{zcm}{}{}%
  {\UseTextSymbol{T1}{\textsterling}}
```

In order to make this particular piece of code work, the font designer would of course need to set up an entire family of T1-encoded fonts in parallell with the main T1R-encoded font family. Alternatively, one could instead collect all these miscellaneous glyphs from an entire family in one font (that would probably become U-encoded), but then one needs to define a lot more variants. In any case, this trick of implementing a command as using a glyph in another font cannot be used for '¡' and '¿', as these need to be accessible through syntactic ligatures as well. What's more, font designers to be should be aware that one cannot have kerns between two glyphs in different fonts.

A list of the accenting commands of the T1R encoding can be found in Table 7. It is hardly exciting, but the font designer should find the information in the 'Remark' column interesting; it lists all the compositions of the accenting commands and whether they are implemented as compositions of the main command or compositions of some variant.

1.5 On hyphenation patterns for the T1R encoding

TeX is constructed so that the hyphenation patterns must match the coding scheme of the font used for the text that is to be hyphenated. It is therefore possible that hyphenation patterns made for use with the T1 encoding does not always work with the T1R encoding. They will work, however, with a font whose coding scheme deviates from that of the T1 encoding only in that some ligatures (or possibly some symbols that never occur as part of a word) has replaced some letters. This is because when TeX hyphenates a word, it treats a ligature as the sequence of letters it is composed from and not as the single character it may be in the font. It does not matter whether there is a hyphenation pattern mensioning character n or not when character n occurs in the output only as a ligature of other characters.

Even in other cases, there is a good chance that hyphenation patterns made for use with the T1 encoding will work without modification for a font with the T1R encoding. The main reason for this is that one usually does not have hyphenation patterns involving letters of which one does not intend to form automatically hyphenatable words active. If one wants hyphenation of words containing a certain character, one also wants it to have a slot of its own in the coding scheme of the font. In most cases, this means that the character is in the same slot as in the T1 encoding.

2 Implementation

First the file announces itself.

```
⟨*encoding⟩
\ProvidesFile{t1renc.def}
   [1998/12/17 Relaxed TeX latin text encoding, version 1.00 (beta)]
```

Then there is a check for if the relenc package has been read. If it hasn't, there is no point in contiuing.

```
\@ifundefined{RE@text@variable}{%
   \PackageError{T1R encoding}{%
      The definition of the T1R encoding requires that\MessageBreak
      the ‘relenc’ package is loaded first}%
   {The T1R encoding cannot be defined.\MessageBreak
      If you continue, you will most likely face further errors.%
      \MessageBreak The best option is to type ‘x’ and fix your
      manuscript.}%
   \endinput
}{}
```

The encoding declares itself and its font substitution.

```
\DeclareFontEncoding{T1R}{}{}
\DeclareFontSubstitution{T1R}{zcm}{m}{n}
```

The search path is set; this is something only relaxed encodings do. The most noteworthy point about this search path is that it includes searching for definitions from the T1 encoding. Thanks to this, the T1R encoding can do without definitions of its own for many variable commands (saving some memory), so the lines defining these have been commented out below.

```
\SetEncodingSearchPath{T1R}{
   {\cf@encoding/\f@family/\f@series/\f@shape}
   {\cf@encoding/\f@family/?/\f@shape}
   {\cf@encoding/\f@family/?/?}
   {\cf@encoding/?/?/?}
   {\RE@convert@nfss{T1}}
   {\RE@convert@nfss{?}}
}
```

The accenting commands are declared. I have choosen to use the same default definitions as in the T1 encoding, despite the fact that the definitions used for \c and \k do not always do what they should (accents are positioned in curious places). These commands are declared to be variable though, so a font designer can override them with ones that are better suited for the font family in question.

```
\DeclareTextAccent{\`}{T1R}{0}
\DeclareTextAccent{\'}{T1R}{1}
\DeclareTextAccent{\^}{T1R}{2}
\DeclareTextAccent{\~}{T1R}{3}
\DeclareTextAccent{\"}{T1R}{4}
\DeclareTextVariableAccent{\H}{T1R}{5}
\DeclareTextAccent{\r}{T1R}{6}
```

```
\DeclareTextVariableAccent{\v}{T1R}{7}
\DeclareTextVariableAccent{\u}{T1R}{8}
\DeclareTextAccent{\=}{T1R}{9}
\DeclareTextVariableAccent{\.}{T1R}{10}
\DeclareTextCommand{\b}{T1R}[1]
  {{\o@lign{\relax#1\crcr\hidewidth\sh@ft{29}%
    \vbox to.2ex{\hbox{\char9}\vss}\hidewidth}}}
\DeclareTextVariableCommand{\c}{T1R}[1]
  {\setbox\z@\hbox{#1}\ifdim\ht\z@=1ex\accent11 #1%
    \else{\ooalign{\hidewidth\char11\hidewidth
       \crcr\unhbox\z@}}\fi}
\DeclareTextCommand{\d}{T1R}[1]
  {{\o@lign{\relax#1\crcr\hidewidth\sh@ft{10}.\hidewidth}}}
\DeclareTextVariableCommand{\k}{T1R}[1]
  {\oalign{\null#1\crcr\hidewidth\char12}}
```

The symbol commands are declared. These are pretty straightforward, but *SS* is a bit of a special case. The command `\SS` must exist as the upper case equivalent of `\ss`, but is there any case where the *SS* glyph is different from two S's?[2]

```
\DeclareTextSymbol{\AE}{T1R}{198}
\DeclareTextSymbol{\DH}{T1R}{208}
\DeclareTextSymbol{\DJ}{T1R}{208}
\DeclareTextVariableSymbol{\L}{T1R}{138}
\DeclareTextVariableSymbol{\NG}{T1R}{141}
\DeclareTextSymbol{\OE}{T1R}{215}
\DeclareTextSymbol{\O}{T1R}{216}
\DeclareTextVariableSymbol{\SS}{T1R}{223}
\DeclareTextSymbol{\TH}{T1R}{222}
\DeclareTextSymbol{\ae}{T1R}{230}
\DeclareTextSymbol{\dh}{T1R}{240}
\DeclareTextVariableSymbol{\dj}{T1R}{158}
\DeclareTextSymbol{\guillemotleft}{T1R}{19}
\DeclareTextSymbol{\guillemotright}{T1R}{20}
\DeclareTextSymbol{\guilsinglleft}{T1R}{14}
\DeclareTextSymbol{\guilsinglright}{T1R}{15}
\DeclareTextSymbol{\i}{T1R}{25}
\DeclareTextSymbol{\j}{T1R}{26}
\DeclareTextVariableSymbol{\l}{T1R}{170}
\DeclareTextVariableSymbol{\ng}{T1R}{173}
\DeclareTextSymbol{\oe}{T1R}{247}
\DeclareTextSymbol{\o}{T1R}{248}
\DeclareTextSymbol{\quotedblbase}{T1R}{18}
\DeclareTextSymbol{\quotesinglbase}{T1R}{13}
\DeclareTextSymbol{\ss}{T1R}{255}
\DeclareTextSymbol{\textasciicircum}{T1R}{`\^}
\DeclareTextSymbol{\textasciitilde}{T1R}{`\~}
```

[2] I am currently thinking about removing the *SS* glyph from the required coding scheme, so if anyone has any opinions on this particular matter, please share them with me.

```
\DeclareTextSymbol{\textbackslash}{T1R}{`\\}
\DeclareTextSymbol{\textbar}{T1R}{`\|}
\DeclareTextSymbol{\textbraceleft}{T1R}{`\{}
\DeclareTextSymbol{\textbraceright}{T1R}{`\}}
\DeclareTextSymbol{\textcompwordmark}{T1R}{23}
\DeclareTextSymbol{\textdollar}{T1R}{`\$}
\DeclareTextSymbol{\textemdash}{T1R}{22}
\DeclareTextSymbol{\textendash}{T1R}{21}
\DeclareTextVariableSymbol{\textexclamdown}{T1R}{189}
\DeclareTextSymbol{\textgreater}{T1R}{`\>}
\DeclareTextSymbol{\textless}{T1R}{`\<}
\DeclareTextCommand{\textperthousand}{T1R}{\%\char 24 }
\DeclareTextCommand{\textpertenthousand}{T1R}{\%\char 24\char 24 }
\DeclareTextVariableSymbol{\textquestiondown}{T1R}{190}
\DeclareTextSymbol{\textquotedblleft}{T1R}{16}
\DeclareTextSymbol{\textquotedblright}{T1R}{17}
\DeclareTextSymbol{\textquotedbl}{T1R}{`\"}
\DeclareTextSymbol{\textquoteleft}{T1R}{`\`}
\DeclareTextSymbol{\textquoteright}{T1R}{`\'}
\DeclareTextVariableSymbol{\textsection}{T1R}{159}
\DeclareTextVariableSymbol{\textsterling}{T1R}{191}
\DeclareTextSymbol{\textunderscore}{T1R}{95}
\DeclareTextSymbol{\textvisiblespace}{T1R}{32}
\DeclareTextSymbol{\th}{T1R}{254}
```

The last thing to declare are all the compositions. It starts with the compositions from block 2 (slots 128–191), which are all in some way variable.

128 = "80.

```
\DefineTextVariantComposition{\.}{T1R}{}{}{}{i}{`\i}
\DefineTextVariantComposition{\u}{T1R}{}{}{}{A}{128}
\DefineTextVariantComposition{\k}{T1R}{}{}{}{A}{129}
\DeclareVariableTextComposition{\'}{T1R}{C}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{C}{130}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{C}{131}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{D}{132}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{E}{133}
\DefineTextVariantComposition{\k}{T1R}{}{}{}{E}{134}
\DefineTextVariantComposition{\u}{T1R}{}{}{}{G}{135}
```

136 = "88.

```
\DeclareVariableTextComposition{\'}{T1R}{L}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{L}{136}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{L}{137}
\DeclareVariableTextComposition{\'}{T1R}{N}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{N}{139}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{N}{140}
\DefineTextVariantComposition{\H}{T1R}{}{}{}{O}{142}
\DeclareVariableTextComposition{\'}{T1R}{R}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{R}{143}
```

144 = "90.

```
\DefineTextVariantComposition{\v}{T1R}{}{}{}{R}{144}
\DeclareVariableTextComposition{\'}{T1R}{S}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{S}{145}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{S}{146}
\DeclareVariableTextComposition{\c}{T1R}{S}
% \DefineTextCompositionVariant{\c}{T1R}{}{}{}{S}{147}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{T}{148}
\DeclareVariableTextComposition{\c}{T1R}{T}
% \DefineTextCompositionVariant{\c}{T1R}{}{}{}{T}{149}
\DefineTextVariantComposition{\H}{T1R}{}{}{}{U}{150}
\DeclareVariableTextComposition{\r}{T1R}{U}
% \DefineTextCompositionVariant{\r}{T1R}{}{}{}{U}{151}
```

152 = "98.

```
\DeclareVariableTextComposition{\"}{T1R}{Y}
% \DefineTextCompositionVariant{\"}{T1R}{}{}{}{Y}{152}
\DeclareVariableTextComposition{\'}{T1R}{Z}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{Z}{153}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{Z}{154}
\DefineTextVariantComposition{\.}{T1R}{}{}{}{Z}{155}
\DefineTextVariantComposition{\.}{T1R}{}{}{}{I}{157}
```

160 = "A0.

```
\DefineTextVariantComposition{\u}{T1R}{}{}{}{a}{160}
\DefineTextVariantComposition{\k}{T1R}{}{}{}{a}{161}
\DeclareVariableTextComposition{\'}{T1R}{c}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{c}{162}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{c}{163}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{d}{164}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{e}{165}
\DefineTextVariantComposition{\k}{T1R}{}{}{}{e}{166}
\DefineTextVariantComposition{\u}{T1R}{}{}{}{g}{167}
```

168 = "A8.

```
\DeclareVariableTextComposition{\'}{T1R}{l}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{l}{168}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{l}{169}
\DeclareVariableTextComposition{\'}{T1R}{n}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{n}{171}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{n}{172}
\DefineTextVariantComposition{\H}{T1R}{}{}{}{o}{174}
\DeclareVariableTextComposition{\'}{T1R}{r}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{r}{175}
```

176 = "B0.

```
\DefineTextVariantComposition{\v}{T1R}{}{}{}{r}{176}
\DeclareVariableTextComposition{\'}{T1R}{s}
% \DefineTextCompositionVariant{\'}{T1R}{}{}{}{s}{177}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{s}{178}
\DeclareVariableTextComposition{\c}{T1R}{s}
% \DefineTextCompositionVariant{\c}{T1R}{}{}{}{s}{179}
```

```
\DefineTextVariantComposition{\v}{T1R}{}{}{}{t}{180}
\DeclareVariableTextComposition{\c}{T1R}{t}
% \DefineTextCompositionVariant{\c}{T1R}{}{}{}{t}{181}
\DefineTextVariantComposition{\H}{T1R}{}{}{}{u}{182}
\DeclareVariableTextComposition{\r}{T1R}{u}
% \DefineTextCompositionVariant{\r}{T1R}{}{}{}{u}{183}
```

184 = "B8.

```
\DeclareVariableTextComposition{\"}{T1R}{y}
% \DefineTextCompositionVariant{\"}{T1R{}{}{}}{y}{184}
\DeclareVariableTextComposition{\'}{T1R}{z}
% \DefineTextCompositionVariant{\'}{T1R{}{}{}}{z}{185}
\DefineTextVariantComposition{\v}{T1R}{}{}{}{z}{186}
\DefineTextVariantComposition{\.}{T1R}{}{}{}{z}{187}
```

Here follows the compositions in block 3 (slots 192–255). They are exactly the same as in the T1 encoding.

192 = "C0.

```
\DeclareTextComposite{\`}{T1R}{A}{192}
\DeclareTextComposite{\'}{T1R}{A}{193}
\DeclareTextComposite{\^}{T1R}{A}{194}
\DeclareTextComposite{\~}{T1R}{A}{195}
\DeclareTextComposite{\"}{T1R}{A}{196}
\DeclareTextComposite{\r}{T1R}{A}{197}
\DeclareTextComposite{\c}{T1R}{C}{199}
```

200 = "C8.

```
\DeclareTextComposite{\`}{T1R}{E}{200}
\DeclareTextComposite{\'}{T1R}{E}{201}
\DeclareTextComposite{\^}{T1R}{E}{202}
\DeclareTextComposite{\"}{T1R}{E}{203}
\DeclareTextComposite{\`}{T1R}{I}{204}
\DeclareTextComposite{\'}{T1R}{I}{205}
\DeclareTextComposite{\^}{T1R}{I}{206}
\DeclareTextComposite{\"}{T1R}{I}{207}
```

208 = "D0.

```
\DeclareTextComposite{\~}{T1R}{N}{209}
\DeclareTextComposite{\`}{T1R}{O}{210}
\DeclareTextComposite{\'}{T1R}{O}{211}
\DeclareTextComposite{\^}{T1R}{O}{212}
\DeclareTextComposite{\~}{T1R}{O}{213}
\DeclareTextComposite{\"}{T1R}{O}{214}
```

216 = "D8.

```
\DeclareTextComposite{\`}{T1R}{U}{217}
\DeclareTextComposite{\'}{T1R}{U}{218}
\DeclareTextComposite{\^}{T1R}{U}{219}
\DeclareTextComposite{\"}{T1R}{U}{220}
\DeclareTextComposite{\'}{T1R}{Y}{221}
```

224 = "E0.

```
\DeclareTextComposite{\`}{T1R}{a}{224}
\DeclareTextComposite{\'}{T1R}{a}{225}
\DeclareTextComposite{\^}{T1R}{a}{226}
\DeclareTextComposite{\~}{T1R}{a}{227}
\DeclareTextComposite{\"}{T1R}{a}{228}
\DeclareTextComposite{\r}{T1R}{a}{229}
\DeclareTextComposite{\c}{T1R}{c}{231}
```

232 = "E8.

```
\DeclareTextComposite{\`}{T1R}{e}{232}
\DeclareTextComposite{\'}{T1R}{e}{233}
\DeclareTextComposite{\^}{T1R}{e}{234}
\DeclareTextComposite{\"}{T1R}{e}{235}
\DeclareTextComposite{\`}{T1R}{i}{236}
\DeclareTextComposite{\`}{T1R}{\i}{236}
\DeclareTextComposite{\'}{T1R}{i}{237}
\DeclareTextComposite{\'}{T1R}{\i}{237}
\DeclareTextComposite{\^}{T1R}{i}{238}
\DeclareTextComposite{\^}{T1R}{\i}{238}
\DeclareTextComposite{\"}{T1R}{i}{239}
\DeclareTextComposite{\"}{T1R}{\i}{239}
```

240 = "F0.

```
\DeclareTextComposite{\~}{T1R}{n}{241}
\DeclareTextComposite{\`}{T1R}{o}{242}
\DeclareTextComposite{\'}{T1R}{o}{243}
\DeclareTextComposite{\^}{T1R}{o}{244}
\DeclareTextComposite{\~}{T1R}{o}{245}
\DeclareTextComposite{\"}{T1R}{o}{246}
```

248 = "F8.

```
\DeclareTextComposite{\`}{T1R}{u}{249}
\DeclareTextComposite{\'}{T1R}{u}{250}
\DeclareTextComposite{\^}{T1R}{u}{251}
\DeclareTextComposite{\"}{T1R}{u}{252}
\DeclareTextComposite{\'}{T1R}{y}{253}
⟨/encoding⟩
```

References

[1] Alan Jeffrey, Rowland McDonnell (manual), Sebastian Rahtz, Ulrik Vieth: *The fontinst utility* (v 1.8), `fontinst.dtx`, in CTAN at `ftp://ftp.tex.ac.uk/tex-archive/fonts/utilities/fontinst/`...