A package for Church-Slavonic typesetting

Andrey Slepuhin

pooh@shade.msu.ru

1 Introduction

The multilingual ability of TEX is one of its most important properties. Due to TEX it became possible to produce high-quality books in many different languages (sometimes with very exotic grammatic rules). For more than 10 years of its existence TEX became a real polyglot and it seems that it doesn't want to stop evaluating. In this paper one more, may be rather exotic, example of practical usage of TEX is considered, and also many ideas and solutions which result from 5-year experience of TEX using.

2 General solutions

How does a language-specific package have to look like from the point of view of a computer publishing system? It must include at least the following components:

- quality fonts;
- tools for simplifying the text formatting;
- hyphenation table;
- punctuation or some other poligraphic rule description;

These requirements became a basis to СлЪTEX development. The two first items got quite satisfactory realization. As to the realization of the third one – it depends on the volume of the dictionary, which is not sufficiently complete yet. The fourth item is absent because the Church-Slavonic language has no precise rules of punctuation or whatever similar things.

3 Fonts

Designing the quality-fonts is, in general, a very hard task and moreover the author's knowledge on this subject at the beginning of this work were minimal. So, the designing

of the base version of fonts took more than half a year, and different improvements are still under development. Among the factors that made the work more complicated, a large number of symbols (only letters – 44) in the Church-Slavonic alphabet should be noted. Also the glyphs of symbols have a very few similar elements. The typeface, which had a wide spreading at the beginning of the XX century, was taken as a model of created fonts. The following technology was applied for the fonts developing: the symbols were magnified and separate elements were extracted, then base and control points of outline curves were placed manually and the METAFONT macros were designed; the obtained symbols were finally improved using the METAFONT graphic output.

4 The diacritical signs problem

The main problem, that occurred during the СлѣТEX development, was connected with the fact, that every word in a Church-Slavonic texts has at least one diacritical sign. None of computer publishing systems (except TEX, of course), known to the author, contains any convenient tools for typesetting a text with accents. TEX uses `\accent` macro for this purpose, but this macro seems to be designed for rather rare usage, because it gives the following undesirable effects:

- the kern between accented and previous symbols disappear;
- TEX doesn't make any hyphens in the remainder of the word after accented symbol and can make invalid hyphens in the initial part of the word;

These effects are arising because TEX uses explicit kern while expanding `\accent` macro. So, it seems, that the best solution of the diacritical signs problem (realized for many European languages, for example) is a method, when a letter together with an accent is represented by a single character in the font. However, in the case of the Church-Slavonic language this solution cannot be applied in proper form, because there are too many possible pairs 'letter – accent', and a limit of 256 symbols will be exhausted.

It would be wonderful if the following idea works: several pairs 'letter – accent' have positions equal modulo 256, and their metrics are identical. Unfortunately, it's impossible to force TEX to put a symbol with character code greater than 256 into DVI-file. Such restriction is especially misunderstanding, because the DVI-file format supports the usage of symbols with character codes up to $2^{32} - 1$. One more well-known method to deal with the accents is their realization as strongly shifted left characters of zero width. Such a variant is unsatisfactory too, because it does not solve the kerning problem and significantly complicate the hyphenation table constructing.

To solve the accent problem we need to understand how the diacritical signs in Church-Slavonic language are placed. It can be found, that some of them can be placed only over the first letter in the word, and some can be placed only over the last letter. These two cases are realized by special macros `\fcaccent` and `\lcaccent`. The last macro can be written in a very simple way, because it needs only to locate the accent

with the help of kerns. The macro `\fcaccent` has `\nobreak\hskip0pt` construction in addition, which enables the hyphenation of the word after the diacritical sign.

As for the accents in the middle of a word, some of them are realized together with the corresponding letters, and other are representing symbols, used, in general, for abbreviation of certain words (in Church-Slavonic language they are called 'titlo'). The words, containing these symbols, as a rule, cannot be hyphenated, so it became possible to write a special macro, placing an accent and preserving kern both before and after the symbol. It is a quite sophisticated macro, which use such powerful TEX tool as `\futurelet`. The text of this macro is given below:

```
\def\caccent#1#2{%
#2\setbox2=\hbox{#2}\setbox1=\hbox{#1}%
\dimen0=\ht2\advance\dimen0 by -1ex%
\dimen1=\wd1\advance\dimen1 by \wd2%
\divide\dimen1 by 2%
\kern-\dimen1\raise\dimen0\hbox{#1}%
\advance\dimen1 by -\wd1%
\kern\dimen1%
\def\tmp{\explkern{#2}\next@}%
\futurelet\next@\tmp%
}%
```

The `\explkern` macro simply adds the kern, that must be placed between its arguments:

```
\def\explkern#1#2{%
\def\next@@{}%
\ifcat#2a%
\explkern@#1#2\else%
\ifcat#2.%
\explkern@#1#2\else%
\ifx#2\-%
\explkern@#1#2\else%
\fi\fi\fi%
}%
\def\explkern@#1#2{%
\setbox0=\hbox{#1#2}%
\setbox1=\hbox{{#1}{#2}}%
\dimen1=\wd0\advance\dimen1by-\wd1%
\kern\dimen1%
}%
```

For the convenience of the text typesetting, the symbols `'`, `"`, `` ` ``, `~`, `_`, `|` and `<` are made active and are expanded to the corresponding macros. The selection of Church-Slavonic

mode is realized by the `\beginslav` macro, and return to usual mode is realized by the `\endslav` macro.

5 Slide making

Another problem, that come into consideration during package development is the problem of slide making. Almost all Church-Slavonic texts are two-colored. For the implementation of the color separation and for obtaining separate slides for each color, the SliTEX's idea of using 'invisible' fonts was applied. However, kerning problems make impossible the usage of pure SliTEX. Indeed, having a word with a first letter emphasized by another color (in Church-Slavonic texts it occurs very often), SliTEX looses the required kern between the first letter and the remainder of the word when switching to another font. So, for such cases we need special macros. To implement the color separation a special font selection scheme was designed, slightly similar to NFSS. After including the font description file and appropriate macros, user can declare usage of any color via the macro `\newcolor(<color>)`. This macro induces the macros `\<color>g{<any text>}` and `\<color>`. The first of them switches the color, preserving kern, and the second switches it without preserving any implicit kern (TEX interprets this macro in the simplest way, so its usage is approved). Now, typing `\showcolor(<color>)` or `\hidecolor(<color>)` in the input file, we can make any selection by a specified color visible or invisible in output. The text before the first usage of `\<color>g{<any text>}` or `\<color>` will be always visible.

6 Numeration

In Church-Slavonic language a literal numeration is accepted, which can be described by the following algorithm:

Given an integer $n \geq 0$. Let $S(n)$ be its representation in Church-Slavonic language. See also Table 1.

The representation of zero is absent in Church-Slavonic language, but let it be empty for conveniency.

If $10 \leq n < 20$, then $S(n) = S(n \bmod 10)S(10)$. If $20 \leq n < 100$, then $S(n) = S(n - (n \bmod 10))S(n \bmod 10)$. If $100 \leq n < 1000$, then $S(n) = S(n - (n \bmod 100))S(n \bmod 100)$. If $1000 \leq n < 10000$, then $S(n) = \not S(n - (n \bmod 1000)S(n \bmod 1000)$.

There are disagreements about representation of numbers greater than 9999, and by this reason it is not implemented yet. The macro `\slnum(<number>)` automatically generates the number representation in the Church-Slavonic language. For example, `\slnum(1995)` gives ѧ҂ацѱїѕ. One would be careful, because this macro is valid only in Church-Slavonic mode.

n	$S(n)$ n	$S(n)$	n	$S(n)$	
1	ѧ	10	і	100	ҁ
2	ѩ	20	ѯ	200	҂
3	ѳ	30	ѵ	300	҃
4	ѫ	40	ѷ	400	҄
5	ѣ	50	ѱ	500	҅
6	ѕ	60	ѹ	600	҆
7	ҙ	70	ѻ	700	҇
8	ѥ	80	ѽ	800	҈
9	ѯ	90	ѿ	900	҉

Table 1: Numeration in Church-Slavonic language

7 TEX without encoding

During the work on ЄѩѣTEX an idea appeared which allows to solve the compatibility problem while transferring any package to another platform. This problem is especially actual in Russia, because Russian letter encodings on different platforms do not coincide. A version of TEX cyrillisation made by CyrTUG is specific for PC-compatible computers under MS DOS. It causes, in particular, the disgust of numerous UNIX users in big research institutes, which need TEX most of all.

The idea of easy transferring any TEX package to different platforms is given below:

- The encoding table containing a map between character codes and their symbolic names (for example, like PostScript names) must be defined for each specific platform and font family.
- The certain utility must be written (it can be done even by TEX!) which generates two files: TEX encoding table and METAFONT encoding table, from the original one.
- The set of METAFONT macros must be added to redefine `beginchar` macro; it must allow the usage of symbolic names instead of character codes by declaring `usenames:=1` or whatever like this.
- Hyphenation table must be written, using the symbolic names; when generating base file, firstly TEX should read encoding, then it should convert original hyphenation table into temporary file using current encoding and then it should read the file obtained.
- `\catcode`, `\lccode` and `\uccode` should be defined using symbolic names.

This idea is implemented in the last version of the package represented and is now in the process of testing. It should be hoped that new CyrTUG's cyrillisation versions will be written in the form described above. It would facilitate the work for many TEX users in Russia and for people who need to typeset Russian (or other Cyrillic) texts.

8 Type 1 from Metafont?

The one more idea, implemented as a part of СлЂTEX project, came from the article[3]. Its realization was forced by appearing a PostScript-printer on the author's table. There was written a set of METAFONT macros which allow to obtain a text representation of Type 1 fonts from METAFONT sources and, furthermore, a downloadable font by L. Hetherington's Type 1 utilities. This macro package initially was designed to solve the specific problem of representation Church-Slavonic in Type 1 format, but the conversion of Computer Modern fonts (and others) is also possible. This work requires a special consideration and is not described in the paper.

9 Problems and plans

The most important of the update problems is the adaptation of СлЂTEX package to LaTeX2ε. When this paper was being written, the author had the distribution of LaTeX2ε for a month, and this distribution was not installed due to the lack of the time. Another problem is connected with the fact that the current font version contains only symbols of modern Church-Slavonic language, whereas symbols from ancient versions of language are often needed. The author also plans to develop a package for typesetting music in non-linear notation (so-called 'krjuki') being in use before XVIIIth century. The following problems connected to Church-Slavonic typesetting also would be noted: designing a font of initial caps and a special font for headings. In this font different combinations of letters must have specific glyphs (this task seems to be a little fantastic, because such font should have a monstrous number of symbols and ligatures).

It would be hoped that somebody shares the author's interest in the problem of Church-Slavonic and ancient texts. May be sometime a multilingual edition of Bible (in Church-Slavonic, Greek, Latin, Hebrew ... what else?) made by TEX come in appearance.

10 Examples

A simple example of a Church-Slavonic text:

```
г|сди <i_исе х|срт‘е, с_не б_жій,
пом’илуй м‘я гр’эшнаго
```

and the result of its compilation:

гд҃и їйсе хр҃тѣ, сйе бжїй, помúлуй мѧ грѣшнаго

An example of color separation: a sequence of macros

```
\beginslav\family(slav)\size(12)%
\def\pray{%
\redg Г|с{ди} <i_исе х|срт‘е, с_не
б_жiй, пом’илуй м‘я гр’эшнаго
}%
\black%
\showcolor(red)%
\par\noindent\pray
\hidecolor(red)%
\showcolor(black)%
\par\noindent\pray
\showcolor(red)%
\par\noindent\pray
\endslav
```

gives the result

Г
гд҃и їйсе хр҃тѣ, сйе бжїй, помúлуй мѧ грѣшнаго
Гд҃и їйсе хр҃тѣ, сйе бжїй, помúлуй мѧ грѣшнаго

This example shows that accents can be placed over a group of symbols, not only over a single symbol.

References

[1] Donald E. Knuth. *The TEXbook*. Addison Wesley, Reading, MA, 1990.

[2] Donald E. Knuth. *The METAFONTbook*. Addison Wesley, Reading, MA, 1990.

[3] Bogusław Jackowski, Marek Ryćko. Labyrinth of METAFONT paths in outline. *EuroTEX Proceedings, 1994: 18–32.*

[4] Ieromonakh Alipiy (Gamanovich). *Grammatika tserkovno-slavjanskogo jazyka*. Palomnik, Moscow, 1991.

[5] *Slovar' russkogo jazyka XI–XVII vv*. Nauka, Moscow, 1975.