The **universal** Font

Version 2.0

Christian Holm*

August 1 1998

**Abstract**

This is my implementation of Herbert Bayer’s “universal” font in META-FONT for \TeX and \LaTeX. Extensive support for \LaTeX is supplied. This font is in no way intended to be a correct, not to mention a complete implementation of Herbert Bayer’s original design. This document describes how to use the font with \LaTeX, and also the source code for the characters of the font.

**Contents**

1 Introduction 2

2 The Font and it’s History 3

3 This METAfont Implementation 5

3.1 The METAfont version versus the Original ................. 5

3.2 Features of the font . .................................. 6

3.2.1 Series, Shapes, Sizes, and Special Characters .......... 6

3.2.2 File Names for the METAfont files .................... 7

3.3 The \LaTeX and NFSS Support .......................... 8

3.3.1 Options to uni .................................... 9

3.3.2 Font Selection Commands ............................ 9

3.3.3 Special Character Commands .......................... 11

3.3.4 Other Commands .................................... 11

3.3.5 File Names for the \LaTeX files ....................... 12

3.4 The Major differences between version 1.0 and 2.0 ....... 12

4 \LaTeX Support Files 13

4.1 The style file — uni.sty ............................... 13

4.1.1 Initializing ...................................... 13

4.1.2 Options .......................................... 14

4.1.3 Special Characters ................................. 14

4.1.4 Font Selection ..................................... 17

4.2 The Font Definition Files — *uni.fd* .......................... 20

* Niels Bohr Institute of Physics, Institute of Philosophy, Rethorics, and Education; University of Copenhagen; Denmark; E-mail: <cholm@nbi.dk>
5 The Font Macros and Programs 24
5.1 Base file — unibase.mf ........................ 24
  5.1.1 Font macros ................................ 25
  5.1.2 Control macros ............................ 29
  5.1.3 Drawing macros ............................ 30
5.2 The Minuscules (lowercase letters) — unilow.mf .......... 33
5.3 The Majuscules (uppercase letters) — uniupp.mf ......... 40
5.4 The Numbers — (unidig.mf) .................... 46
5.5 Specials — unospe.mf .......................... 49
5.6 Punctuations — unipun.mf ........................ 51
5.7 Accents — uniacc.mf ............................ 55
5.8 Ligatures — unilig.mf ........................... 57
5.9 Extras — uniext.mf ............................. 59

6 Font driver files 66

7 Commands for the Documentation — unidoc.sty 67
  7.1 Initializing ................................... 67
  7.2 Macros for the Index .......................... 68
  7.3 Descriptive Environments ...................... 69
  7.4 Useful Commands ................................ 70
    7.4.1 The Font Charts ........................... 71
  7.5 Parameters and Miscellaneous Commands ............. 73

A Solution to the \bauhausforms problem 75
  A.1 The Problem ................................... 75
  A.2 Joseph Collins’ Solution ........................ 76

B Copyrights — GNU General Public License 77

C Wishlist 77

D Font Charts 79

1 Introduction

This package contains the METAFONT source and driver files for the “universal” font, designed by Herbert Bayer, a teacher at the Bauhaus school in Weimar, plus a L\TeX package to utilize this font, along with a number of font definition files, as required by the New Font Selection Scheme (NFSS).

About the Documentation

The full documentation of this font is rather large, more then 80 pages actually. Most of it is the programs for the characters for this font, so if you don’t know the METAFONT language, or don’t care how the character are created, you should insert \OnlyDescription into the preamble of uni.dtx.

Notice, that METAFONT macros are not indexed in this documentation. This is because it would take a major rewrite of the doc package to do so, and I really didn’t want to do that.
Some of the macros of doc has been redefined, and if you want to see which, or how I generally did the documentation, please refer to appendix 7.

If you in some way are unsatisfied with some of the characters of the font, do please read the documentation of that character. There may be some notes on why the character looks the way it does. Some of the comments, however are a bit silly and should be skipped at high speed (pretend there is a conditional that says if not silly ... fi).

This Version

This new version of the universal font, provides a number of new features and improvements, both to the font itself and to the \LaTeX{} support macros. Also, a number of corrections has been made to the font programs.

The reason why I jumped one whole version number from 1.0 to 2.0, is that I have taken a whole new approach to the font programs and shapes of the universal font, plus I decided to put some more effort into the \LaTeX{} support.

Thanks and Other Stuff

To those of you who have had the (mixed) pleasure of using version 1.0 of this font, one of the most notaticable changes are to the symbol $\textcircled{A}$. In the old documentation, I complained that I couldn’t find the exact solution to the problem this symbol posed. I also encouraged people to send me any solution they may have had — and guess what — somebody did! Therefore I would like to thanck Joseph Collins for providing me with the solution. If you like Joseph Collins and I like Mathematical puzzles take a look in Appendix A to learn more about this problem and it’s solution.

I also direct your attention to Appendix B for the copyright notice on the universal package (it’s the Gnu General Public Lisence to those of you who know it).

If you in any way have gotten tempted to design yourown font, or to implement some font into METAFONT, I feel obligated to bring you a warning, taken from the METAFONT book by Donald E. Knuth:

Warning: Type design can be hazardous to your other interests. Once you get hooked, you will develop intense feelings about letterforms; the medium will include the message that you read. And you will perpetually be thinking of improvements to the fonts that you see everywhere, especially those of your own design.

2 The Font and it’s History

Bauhaus

The Bauhaus school in germany originally located at Dessau, was a school for any kind of design, ranging from potery to furniture, from painting to — what was considered the prime form of design — architecture. Many famous designers came from, or taught at Bauhaus, for example Mies van der Rhoe, Herbert Bayer, Kandinsky, Walter Gropius and Gerrit Rietveld. The style “die stiil” was explored
here, and painters like Mondrian made large contributions to what today is known as “the Bauhaus style”.

The basic idea of the Bauhaus school was to design items, which along with its aesthetic value, also had a high degree of functionality. Houses were meant to be suited for all kinds of living, while still keeping the beauty that make people glad to see their house. Chairs should be comfortable for their use, as well be able to fit-in in a normal house. All this should be done at a price that made it possible for everybody to own designer-furniture, houses, etc.

The political idea of Bauhaus, was that of a socialist one. Houses are for the people — they have to live in them, and that living should be a good as possible. Therefore the Bauhaus school saw it as its task to provide functional, beautiful everyday items that anybody could afford. Paintings and tapestry shouldn’t hang on museums or art galleries, but in peoples homes, where they would enrich their everyday life.

Herbert Bayer and the “universal” Font

At Bauhaus typography was also studied, not just how written text should be typeset, and how printed characters should look like, but also what the essence of writting is, in it’s practial and design-wise sense. This led Herbert Bayer to formulate some principles of writting:

experiment with simplified way of writing:

1. this way of writing is recommended by all typographic designers as the future way of writing.
2. by writing in minuscles our writing loses nothing, but is easier to read, considerably more economical.
3. why must you for one sound have two tokens, e.g. A and a? why two alphabets for one word, why this double set of signs, when the hall is enough.

herbert bayer 1925

On this principles, Herbert Bayer designed a font, which should have no majuscles (upper case letters), easy to print, and easy to read since it didn’t have any unusefull decoration, but communicated the bare meaning of characters through the simplest forms needed to reconigise a character. This font he called “universal”.

This font contained abolutly no majuscles, since Bayer believed them to be superflous, as it is clear form the quote above.

At the time Herbert Bayer formulated these principles and designed the “universal” font, most printers used Gothic letters, which is allmost overly decorated, so his font ofcourse made contraversy.

Later on, in the 1930’ies, the Bauhaus school drew the attention of Gestapo\(^1\) of Nazi\(^2\) Germany. The school was finally closed in 1936 by Gestapo, because they believed they were promoters of Jewish and Communist culture and propaganda.

\(^1\)Geheime Stats Polizi
\(^2\)Nationalsocializmus
In the aftermath of the closing of the school, most of its ideas were shunted by other designers, and the Bauhaus way of thinking died out. This is properly the reason why Herbert Bayers “universal” font is so little known today.

However, the “universal” font still stands as one of the most compelling developments in font designing. It represents an approach to designing where the functionality is as vital as the expression, and as such I believe it to be one of the most important fonts in the world today.

3 This METAFONT Implementation

This implementation of Herbert Bayers “universal” font, is not supposed to look exactly like the original design. Of course I have tried to the best of my ability to mimic his design as far as I could. However, it is not an easy matter to find a complete, not to mention exact, sample or account of Bayers design.

This implementation is based on the samples I could find, and other implementations of the “universal” font. Many of these other implementations do differ from the original samples, and include characters I couldn’t find in any of the original. So whenever I found disparities, I mainly leaned on the original samples and my understanding of the original design.

3.1 The METAFONT version versus the Original

As mentioned above, Bayer never did design any majuscules for this font, but nonetheless, I have included them into this implementation. This I did, because I think most people will have a hard time writing in minuscules (lowercase letters) alone. Of course, if you agree with Bayer, you should simply not use them.

There are also some other differences, mostly due to the fact, that I never found a complete sample of the original font. The major differences between the original font and this implementation is summarized below:

- **Majuscules:** Majuscules are present, even though they weren’t in the original design.
- **Digits:** These are based on other implementations, and my general conception of the original design.
- **Punctuations:** As above.
- **Accents:** As above.
- **Symbols:** As above.
- **Bauhaus Symbols:** I have added some various symbols I have found in connection with Bauhaus to the font. The reason is I find them beautiful, and I had some space to fill.
- **Numerous Shapes and Weights:** I don’t think Bayer ever designed slanted characters, a bold face version of the font, and he could never have designed a small caps version of the font. However, these are present in this implementation. I included these features, because I believe them to be of general utility, and it makes the font conform more to the Computer Modern Roman font, and NFSS.
3.2 Features of the font

Rather than using cmbase.mf\textsuperscript{3}, and then redefine some macros, I chose to make a new base file myself, i.e., unibase.mf\textsuperscript{4}. This file contains a number of macros\textsuperscript{4} I have used in the character programs.

The macros of unibase.mf actually reflects my conception of the font. There are three basic drawing macros:

\texttt{unicir} which draws a circle,
\texttt{uniarc} which draws a segment (arc) of a circle, and
\texttt{unilne} which draws a straight line.

I believe, that Bayer intended the font to be made of these two basic shapes: the arc, and the line. Also, to keep things simple, and therefore easy to print, all shapes should be of the same thickness, i.e., as if drawn with a pen of equal thickness. I have made one deviation from this, however. All majuscules are drawn with a thicker ‘pen’, which makes the output nicer, I think.

Incidently, this made the programs of the characters much simpler, and shorter.

3.2.1 Series, Shapes, Sizes, and Special Characters

Below is a sample of each series/shape combination available in this implementation of the font, along with the \LaTeX\ commands that drive them:

Medium upright (\texttt{textuni}): \texttt{The dazed brown fox quickly gave 1234-567890 jumps!}

Medium slanted (\texttt{textunisl}): \texttt{The dazed brown fox quickly gave 1234-567890 jumps!}

Medium small caps (\texttt{textunisc}): \texttt{The dazed brown fox quickly gave 1234-567890 jumps!}

Medium strict (\texttt{textunist}): \texttt{The dazed brown fox quickly gave 1234-567890 jumps!}

Bold face upright (\texttt{textunibf}): \texttt{The dazed brown fox quickly gave 1234-567890 jumps!}

Bold face slanted (\texttt{textunibsl}): \texttt{The dazed brown fox quickly gave 1234-567890 jumps!}

Bold face small caps (\texttt{textunibsc}): \texttt{The dazed brown fox quickly gave 1234-567890 jumps!}

Bold face strict (\texttt{textunibst}): \texttt{The dazed brown fox quickly gave 1234-567890 jumps!}

Everyone of these shapes are available in size 8, 9, 10, 12, 17 pt, and \META\-FONT can of course create others.

\textsuperscript{3}Computer Modern Roman base file
\textsuperscript{4}A better name for ‘base’ would be ‘library’, and then the file would be libuni.mf, but to conform to CM, I used ‘base’.
Also a number of non–standard characters are available in the font. Below is a table of these characters along with the \LaTeX commands that drive them.

Notice that "." and "," is present in the table. This is because these characters are not directly defined, but is supplied as ligatures. This can be done, because they are simple doubles of "." and ",".

A quick look on the table will also reveal some characters that generally isn’t present in the standard OT1 encoding\(^5\), but generally present in the T1 encoding\(^6\). I have done this, both to provide an (almost) complete font for the European languages, but also because I anticipate the universal font some time in the future will shift, or at least be available, in the T1 encoding\(^7\). Please note, that in the small caps shaped fonts, \textit{dh} does not give \textit{D}, but a \textit{A}, that is a small caps shaped version of \textit{q}. Also there is no command \textit{varq} defined.

In appendix D is some charts showing the font in different series and shapes.

### 3.2.2 File Names for the METAFONT files

#### Base File and Source Files

The base file and the files containing the code for the characters of the universal font, all starts with \textit{uni}, to reflect the connection of the files. The next five possible letters reflects what kind of code is contained within the file, e.g., the base file ends in \textit{base}, the file containing the code for the minuscules (lower case letters) end in \textit{lower}, and so forth.

#### Font Driver Files

The font driver filenames has been chosen to conform to the \textit{fontname} scheme, because this scheme is used by most \TeX, \LaTeX, and METAFONT systems (anyway those that use \textit{kpathsea}, which is the most).

\(^5\)The OT1 encoding is the 7 bit encoding of the Computer Modern fonts by Donald E. Knuth. 7 bit means it contain 128 (= \textit{2}^7) characters.

\(^6\)The T1 encoding is an encoding especially designed for the (western) European languages. It was founded by the \TeX User’s Group, on a seminar in Cork, and is the basis of the dc fonts. T1 is an 8 bit encoding, which means it has 256 (= \textit{2}^8) characters.

\(^7\)T1 is generally considered the encoding of the future, and in the long term, it is most likely the encoding of \LaTeX3.
The filename all contain the three characters `ful`, where the `f` stands for `public` and `ul` for `universal`.

Next comes a letter which is one of `m` (`medium`) or `b` (`bold`), which represents the series of the font.

Then comes one or two letters, which are `r` (`upright` or roman), `o` (`slanted`, or oblique), `c` (`small caps`), or `st` (`strict`), which represent the shape of the font.

Finally the filename ends with the designsize in points.

Thus the complete syntax for the font driver file names is: The `fontname`

\[
\begin{align*}
\langle \text{filename} \rangle & := \langle \text{supplier} \rangle \langle \text{face} \rangle \langle \text{series} \rangle \langle \text{shape} \rangle \langle \text{size} \rangle . \text{mf} \\
\langle \text{supplier} \rangle & := f \\
\langle \text{face} \rangle & := \text{ul} \\
\langle \text{series} \rangle & := m | b \\
\langle \text{shape} \rangle & := r | o | c | st \\
\langle \text{size} \rangle & := 8 | 9 | 10 | 12 | 17
\end{align*}
\]

scheme actually says to put \langle `encoding` \rangle information after the \langle `shape` \rangle, but since this is `8r` for \TeX{} Text, it would make filenames longer than 8 characters in the cases of \langle `size` \rangle of 10, 12, and 17, so this information is left out (which is permissible in `fontname`, but unfortunate).

This way of naming the font driver files will, if you use `kpathsea`, put the `ful*.pk` files in
\langle `pk–base–dir` \rangle /public/universa/
and the `ful*.tfm` files in
\langle `tfm–base–dir` \rangle /public/universa/
which I think is the intuitively correct place to put them. This also means, that the `*.mf` files provided with this package, should be placed in
\langle `mf–source–base–dir` \rangle /public/universa/
again very intuitive.

Below is a table of the usual directory names under Unix–like and MSDOS–like (including Windows95) systems. \langle `mode` \rangle is dvips’s name for your printer.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Unix–like systems</th>
<th>MSDOS–like systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>\langle <code>tfm–base–dir</code> \rangle</td>
<td>/var/spool/texmf/pk/\langle <code>mode</code> \rangle</td>
<td>C:\FONTS\PK\langle <code>mode</code> \rangle</td>
</tr>
<tr>
<td>\langle <code>tfm–base–dir</code> \rangle</td>
<td>/var/spool/texmf/tfm/</td>
<td>C:\FONTS\TFM\</td>
</tr>
<tr>
<td>\langle <code>tfm–base–dir</code> \rangle</td>
<td>/usr/local/lib/texmf/fonts/source/</td>
<td>C:\TEX\MFINPUTS\</td>
</tr>
</tbody>
</table>

Table 2: Common directory names.

### 3.3 The \LaTeX{} and NFSS Support

To use the \texttt{universal} font with \LaTeX{}2\texttt{e}\textsuperscript{8}, you should load the package `uni` with the command

\texttt{\usepackage[\langle `options` \rangle]{\langle `uni` \rangle}}

in your preamble (i.e., after \texttt{\documentclass} and before \texttt{\begin{document}}). \langle `options` \rangle can be any of the options described below, but no other.

\textsuperscript{8}I have made no attempt to provide support for \LaTeX{}2.09, since this format is obsolete, and those who do use it, will probably never bother to look at CTAN for new fonts anyway.
3.3.1 Options to uni

The strict option is intended to facilitate typesetting of the universal font in a strict bouhaus fashion, that is only in minuscules.

In this font, only the series may be variated, that is, there is a bold series strict shaped font of any size, and a medium series strict shaped font of any size, in the universal family.

This option can be used in conjunction with options medium and bold. Please notice, that it doesn’t make any sense to ask for a small caps or slanted shaped font, while using this option.

Notice that only \textuni and \uni is defined if option strict was given to uni package.

If you give the default option to the uni package, the default font of the document will be universal.

If you also used the option bold the default font will be the universal font in bold series. Otherwise it will be in medium series.

With this option, \textit, \it and \itshape shifts to universal slanted font, i.e., there is no italic font available.

To make it possible to change back to Donald E. Knuth’s Computer Modern Roman font, even when the default option is given, we define macros \textcmr and \cmr, which switches the \fontfamily to cmr locally and globally respectively.

You should use this option with some care, since the universal font isn’t very suited for longer texts, but rather for short letters, quotes, and other pieces of text where the graphical apperance is important.

When this option is given, command \textuni switches to medium series universal font, as do \uni. The other font selection commands behave as always (see below). This is the default option to uni, i.e., not normally needed.

If this option is given, commands \textuni and \uni switches to bold series universal font. Other font selection commands behave as always (see below). Notice that it makes no sense to give both option medium and option bold to the uni package.

3.3.2 Font Selection Commands

These two commands only change the current font family to uni and nothing else.

That means, that if you say for example

\{\al Hello \textuni{world}\}

you get both ‘Hello’ and ‘world’ in slanted shape, and the output would be

Hello world

To put it in another way: Font encoding, shape, size, and baselineskip is preserved under \textuni and \uni, while font family is not.

Exceptions: If you gave the bold option to the uni package, then this command will always give you a bold series font. If you gave the strict option, then this command will always give you a strict shaped font.

These commands can be used in conjunction with \LaTeX commands \textbf, \textsl, and even \sc, or \rm to give different series and shapes.

If you in the previous example intended to shift to medium upright uni-
versol font you could instead have used \textunirm, since this command does not preserve font shape, i.e., the shape is always changed to upright, regardless of the previous shape. So if you said
\begin{verbatim}
\textsl Hello \textunirm{world}
\end{verbatim}
you would get
\begin{verbatim}
Hello world
\end{verbatim}\nActually \textunirm and \unirm isn’t the only commands that aggressively changes most of the font parameters. \textunislo, \textunisc, and \textunist always gives you medium slanted, medium small caps, and medium strict respectively no matter what the values of \f@shape was before.

In the same category is \textuniblf, \textunibsl, \textunibsc, and \textunist which always changes the font series to bold, along with change in shape (upright, slanted, small caps, and strict in that order).
\textunibf All of the ‘aggressive’ commands, do not however change the size and baselineskip of the font. This should be done by using \texttt{\TeX} commands such as \texttt{\small}, \texttt{\Large}, \texttt{\fontsize{⟨size⟩}{⟨lineskip⟩}}, etc.

Notice that the ‘aggressive’ commands always changes to the appropriate font series. That is, even if you gave the bold option to uni, \textunirm will still give you medium upright universal font. This particular instance illustrates the use of the aggressive commands quite well I think.

To summarize: The ‘aggressive’ commands doesn’t preserve family, series and shapes, but does preserve encoding, size, and baslineskip.

\textbf{Warning:} The font shifting commands \textuni... and \uni... in this section is not defined if you gave the strict option to the uni package.

One can also use the rather primitive command \texttt{\unifamily} in conjunction with \texttt{\selectfont} as described in \texttt{\TeX} 2ε \textit{Font Selection}, to change the font family to \texttt{universal} if absolute control is prefered.
\unifamily is used by all the other font switching commands, so if you redefine it, or \texttt{\unifamilydefault}, you could get strange resault.
\unifamilydefault This command normally expands to uni, which is the ‘family’ name of the \universal package. If you redefine this command to be something else, e.g., cmr, \unifamily will load another font.

\texttt{\uniseries} If the bold option to uni is used this command will select the default series of the \universal font, defined in \texttt{\uniseriesdefault}, which ofcourse defaults to b, i.e., bold series. if you redefine \uniseriesdefault to be m, then \uniseries will select medium series fonts.

If bold option wasn’t given, then this expands to nothing, as do \texttt{\uniseriesdefault}. This command is used by \texttt{\textuni} and \texttt{\uni}.

\unishape If the strict option to uni is used this command will select the default shape of the \universal font, defined in \texttt{\unishapedefault}, which ofcourse defaults to st, i.e., strict shape. If you redefine \unishapedefault to be n, then \uniseries will select upright shaped fonts.

If strict option wasn’t given, then this expands to nothing, as do \texttt{\unishapedefault}. This command is used by \texttt{\textuni} and \texttt{\uni}.

\texttt{\stshape} This command switches to strict shape, i.e., all majuscules will be typeset as minuscules. This makes it possible in a simple way to typeset text in the way Herbert Bayer thought it should, as evident from the citation above.

10
This command uses the command \stdefault, which defaults to \st. If you redefine this to something else, e.g., \sl you will get a slanted font.

This command is used by \textunist, \textunibst, \unibst and \unist, and is defined no matter which options you gave to the uni package.

**Warning:** Since strict is a non–standard shape, this command should not be used outside the universal font, since this may give you unexpected results.

### 3.3.3 Special Character Commands

For the individual commands that makes various special characters, please consult table 1 above.

### 3.3.4 Other Commands

\k The macro \k used in the universal font gives the accent ogének, that is a reversed cedilla accent. It takes one argument, which should be a single letter, under which it puts the accent. For example, you could say \textuni{\k{a}} and get à.

Now you can configure the special character commands of the font, via the commands \DeclareUniChar and \DeclareUniCommand. The commands defined via these commands will only work in accordance with it’s definition inside the universal font, and if defined elsewhere according to it’s definition there, else it will give an error message.

\DeclareUniChar is used to define a command sequence representing a single character in the universal font, much like \chardef, though the control sequence will produce an error message outside the universal font, and proper unexpected results outside the OT1 encoding.

\DeclareUniCommand is used to define control sequences inside the universal font, representing many characters or doing complex maneuvers on characters and stuff. The optional argument to \DeclareUniCommand can be used to say how many arguments the control sequence should have, just like \newcommand. However, it is not possible to give a default first argument.

The below definition uses the color package to typeset a square, circle, and triangle in different colours.

\begin{verbatim}
\DeclareUniCommand{\mybauforms}{% 
  \lower.5ex\hbox{\color{blue}\bautriangle}% 
  \kern-.5em\raise.5ex\hbox{\color{read}\baucircle}% 
  \kern-.5em\lower.5ex\hbox{\color{yellow}\bausquare}}
\end{verbatim}

Yet another example, using arguments could be

\DeclareUniCommand{\mybaulogo}{\bauhead\ \Large \#1}

so you could say \mybaulogo{Christian Holm} and get:

\begin{verbatim}
\includegraphics{Christian Holm}
\end{verbatim}

---

9 This should not be a problem.

10 Since the documentation should be available to all, I can not provide you with the outcome of this example, since it needs the color package which may not be available on all sites. I suggest you try it out, or something similar, if you can.
and I bet you can come up with some even more useful and complex commands.

The syntax of \DeclareUniChar and \DeclareUniCommand is

\DeclareUniChar{⟨cmd⟩}{⟨slot⟩}
\DeclareUniCommand{⟨cmd⟩}{⟨arg⟩}{⟨definition⟩}

where ⟨cmd⟩ is the user command defined, ⟨slot⟩ is the number of the character
in the font, ⟨arg⟩ is the number of arguments and ⟨definition⟩ is what ⟨cmd⟩ does.

3.3.5 File Names for the \LaTeX files

All the \LaTeX files contain the three letters uni, to reflect the connectedness of
the files. The font definition files all start with the letter code appropriate for the
encoding.

To follow the scheme of the META\font files, it would be appropriate to place
all \LaTeX files in
⟨tex–base–dir⟩\!/tex/latex/universa/

3.4 The Major differences between version 1.0 and 2.0

First of: a lot of bugs and errors has been corrected. In version 1.0, I had
made the (stupid) mistake of calling the macro mode_unset before I defined
the unsharped units. Of course a quick look in the META\font book showed be
just how stupid this is. This made the font very vulnerable to mode specifications,
which of course isn’t the idea.

Secondly: I chose a completely new approach to the character programs, which
resulted in unibase.mf. The idea is to define a few macros, and then utilize those
in the character programs, so that these programs can be kept simple, efficient,
and intuitive.

A quick look at unibase.mf will also reveal that I chose a new way of adjusting
the characters. This means that the macros bauhaus... present in version 1.0
no longer is needed, and since they only tended to obscure things rather than
simplify them, I went back to the plain META\font macro beginchar, which is
much stronger.

All in all, unibase.mf provides a much stronger and uniform framework for
character design, than did the old universal.mf.

Thirdly: The file names have been kept inside MS\DOS conventions, that is first
name of maximum 8 characters, and last name of maximum 3 characters. This
does mean, however, that some file names are not intuitive, but I have tried to
make them as much as I could.

Also, every file associated with this font, except the font definition files
(*uni.fd), and font driver files (ful*.mf) begins with the three letters uni, to
emphasize the connection.

\footnote{On Unix-like systems ⟨tex–base–dir⟩ is usually something like /usr/local/lib/texmf/, and
on MS\DOS–like systems something like C:\TEX\}. 

12
Fourthly: Some new font shapes are available, as explained above. I found out, during the design of the characters, that the new base file `unibase.mf` kept showing new potentiality, and the extension of the font to include more shapes was very easy inside the frame of this base, so I thought “What the heck!”

Fifthly: Some of the symbols available in version 1.0, has been taken out, and some new, more general characters have been added. Most of the absent symbols where not really of general use, so I decided to take them out, since I was never really satisfied with those anyway. This also made the font contain exactly 128 characters, just like a normal Computer Modern Roman font.

Sixthly: I improved the \LaTeX and NFSS support considerably. The changes are legion, but let me sum up the most important here.

1. Stronger font selection commands.
2. Command names that should be more intuitive.
3. Preparations for \T1 encoding.
4. Conformation to \LaTeX2ε style, and therefore a better chance to conform with the future \LaTeX3 format.
5. More and better options.

4 \LaTeX Support Files

4.1 The style file — uni.sty

In version 2.0, the commands in this file has been redefined using the macros recommended in \LaTeX2ε Font Selection and \LaTeX2ε for Class and Packages Writers. This should make the commands and macros more portable, and secure. Further, it should make it upward–compatible with future releases of other packages and in the end \LaTeX3.

4.1.1 Initializing

First we need to identify the package, its version and release date, etc.

\begin{verbatim}
\def\fileversion{v2.0}
\def\filedate{98/08/01}
\ProvidesPackage{uni}[\filedate\space\fileversion\space universal package.]
\end{verbatim}

Then we setup some new \if commands, to help in different situations, depending on options passed to the package etc.

\begin{verbatim}
\newif\ifuni\unifalse
\newif\ifstrit\stritfalse
\newif\ifult\ultfalse
\newif\ifmedium\mediumfalse
\end{verbatim}
4.1.2 Options

We define some options that can be passed to the package. Option strict is intended to make it possible to do strict Herbert Bayer typesetting, that is only in minuscles. Option default will make the default font universal of the entire document, while option medium and bold decides wether the default font used by \textuni is normal or bold series. Finally option medium is declared default.

10 \DeclareOption{strict}{\stri@ttrue}
11 \DeclareOption{default}{\def@lttrue}
12 \DeclareOption{medium}{\m@diumtrue}
13 \DeclareOption{bold}{\m@diumfalse}
14 \ExecuteOptions{medium}
15 \ProcessOptions\relax

4.1.3 Special Characters

To facilitate the special character positions of the universal font, we define the internal commands \uni@init, and the user commands \DeclareUniChar and \DeclareUniCommand. \DeclareUniChar and \DeclareUniCommand is very much like the \LaTeX commands \DeclareTextSymbol and \DeclareTextCommand, except we have to take into account, that the commands may very well be defined in \otlenc.def or the like. Since some of the characters provided with this font, is usually part of an 8 bit font, I can’t just use the character numbers straight of (this is a 7 bit font), and I have some very non–standard characters as well.

Though the two commands differ a bit, the main mechanism is the same, so I will explain them together, and note the differences.\(^{14}\)

1. When either of the commands is used, we first call the macro \Declare@uni, which sets up 2 or 3 commands.

2. Inside \Declare@uni, we define \temp@a to be \uni@⟨cmd-name⟩, where ⟨cmd-name⟩ is ⟨cmd⟩ stripped of the escape character (\).

3. Then we check wether \cmd is defined.

   • If ⟨cmd⟩ isn’t defined, then we define \temp@c to give \uni@⟨cmd-name⟩ (which isn’t defined yet) inside the universal font, otherwise an error message (\UniError).

   • If ⟨cmd⟩ is defined, we define \temp@b to be \no@uni⟨cmd-name⟩, \temp@c to give \uni@⟨cmd-name⟩ inside the universal font, and \no@uni⟨cmd-name⟩ (which isn’t defined yet) elsewhere. Then we define \no@uni⟨cmd-name⟩ to be what ⟨cmd⟩ currently is.

This finishes \Declare@uni off.

\(^{12}\)Change of \DeclareUniChar on 98/08/01, Version 2.0
\(^{13}\)Change of \DeclareUniCommand on 98/08/01, Version 2.0
\(^{14}\)The syntax of the macroes are explained on page 10.
4. The next step depends a bit on which of the macros that is utilized. What is common is that \texttt{\uni\@\langle cmd-name \rangle (hold in \temp\@a)} is defined to be the last argument of the command.

- In \texttt{\DeclareUniChar}, we define \texttt{\uni\@\langle cmd-name \rangle} via the \TeX{} command \texttt{\chardef} to be \texttt{(slot)}, which should be a number (decimal, octal, or hexal) between 0 and 127. Using the \TeX{} primitive will make \LaTeX{} think of \texttt{(cmd)} as a single character.

- If \texttt{\DeclareUniCommand} was given an optional argument, we use the command \texttt{\DeclareUni@xarg} to define \texttt{\uni\@\langle cmd-name \rangle} — which uses \texttt{\newcommand} with \texttt{one} optional argument — to be \texttt{(definition)}. Otherwise we use \texttt{\DeclareUni@narg} — which uses \texttt{\newcommand} with \texttt{no} optional arguments — to do the trick. Using the \LaTeX{} complex \texttt{\newcommand}, makes \texttt{(cmd)} a strong command.

5. Finally we define \texttt{(cmd)} to be \texttt{\temp\@d}, which was the conditional command defined in step 2.

Notice the use of \texttt{\aftergroup}. This means that \texttt{(cmd-name)} shouldn’t be too long, i.e., > 300 characters, which isn’t a real limitation, but now I said it. This macro, together with the group in \texttt{\temp\@c} is what made it possible to give arguments to \texttt{(cmd)} defined by \texttt{\DeclareUniCommand}.

This way of doing things has a number of benefits. 1) First of I makes it very simple to redefine the behaviour of the \texttt{universal} specific character commands. All you have to do is to use \texttt{\DeclareUniChar} or \texttt{\DeclareUniChar} in your preamble (and \texttt{only} there). 2) Since \TeX{} are faster at evaluating \texttt{\if...'}s, switching to \texttt{universal} commands should take a minimum amount of time. 3) From a portability point of view, this way of shifting meanings of commands such as \texttt{\textquotedblleft}, according to context, doesn’t screw up older documents where the uni package is added to.

There are some disadvantages. 1) Most noticably there is \texttt{no} conformation to \texttt{any} existing encoding, which however is a problem of the in the font, not the \texttt{uni} package. 2) The heavy use of \texttt{\def}, \texttt{\edef}, and \texttt{\let}, is not exactly \LaTeX{}3 policy, but is needed here. 3) The definition of the commands may slow \TeX{} down a bit, but should be bearable.

17 \texttt{\newcommand{\uni@init}{\@unitrue}}
18 \texttt{\DeclareRobustCommand{\Declare@Uni}[1]{}}
19 \texttt{\edef\temp@{\expandafter\@gobble\string#1}}
20 \texttt{\edef\temp@a{\csname uni@\temp@\endcsname}}
21 \texttt{\@ifundefined{\temp@}{}}
22 \texttt{\edef\temp@c{}}
23 \texttt{\noexpand\if@uni\noexpand\aftergroup\temp@a\noexpand\%}
24 \texttt{\else\noexpand\UniError{#1}\noexpand\fi}}{}\%}
25 \texttt{\edef\temp@b{\csname no@uni\temp@\endcsname}}
26 \texttt{\edef\temp@c{}}
27 \texttt{\noexpand\if@uni\noexpand\aftergroup\temp@a\noexpand\%}
28 \texttt{\else\noexpand\aftergroup\temp@b\noexpand\fi}}{}
29 \texttt{\expandafter\let\temp@b=#1}}{}
30 \texttt{\DeclareRobustCommand{\DeclareUniChar}[2]{}}
31 \texttt{\Declare@Uni[#1]}{}
32 \texttt{\expandafter\chardef\temp@a=#2}
33 \texttt{\let\#1\temp@c}
This command takes care of commands for non-standard placement, and symbols
in the universal font, while using standard \LaTeX\ encodings, such as OT1.

For instance, the symbols \textbackslash AD, \textbackslash AC isn't present in Computer Modern fonts, but is
accessible through the command \textbackslash { and } and \textbackslash } in the universal font. On the other
hand \Gamma isn't present in this font, but is replaced by \textbackslash G.

Some of the commands available are well-known to the T1 encoding, and are
provided with the same command names as in that encoding.

The declaration of these commands use my (slick) macros \DcelareUniChar
and \DcelareUniCommand — very neat I think.
4.1.4 Font Selection

\unifamily The real command. This switches to \textit{universal} font, and can be used by authors in conjunction with \LaTeX primitive \texttt{selectfont}. \unifamily is used below in \texttt{textuni}..., and \uni....

\unifamilydefault Here \unifamilydefault is by default defined to be \textit{uni}, but could ofcourse expand to something different. However, the expansion should be a font family (like \textit{cmr}) or the like.

\uniseries If the \texttt{bold} option is used with the \texttt{uni} package, then this command switches to whatever series is defined via \uniseriesdefault (default is \texttt{bold} series), otherwise it does nothing. Since this command is used in \texttt{textuni} and \uni, this way will not allow anyther series if \texttt{bold} option is given, expect \texttt{bold}. However, if \texttt{bold} isn't given, then \texttt{any} series is possible.

\uniseriesdefault Here \uniseriesdefault is defined to be \textit{b} (\textit{medium}) if the \texttt{bold} option was
given to \texttt{uni} package. Otherwise it expands to nothing (\texttt{relax}). This macro is used by \uniseries.

\texttt{Change of \unifamily on 98/08/01, Version 2.0}
\texttt{Change of \uniseries on 98/08/01, Version 2.0}
If the strict option is used with the uni package, then this command switches to whatever shape is defined via \unishapedefault (default is strict shape), otherwise it does nothing. Since this command is used in \textuni and \uni, this way will not allow any other shape if strict option is given, expect strict. However, if strict isn’t given, then any series is possible.

Here \unishapedefault is defined to be st (strict) if the strict option was given to uni package. Otherwise it expands to nothing (\relax). This macro is used by \unishape.

This command selects the shape holded by \stdefault (defaults to st — strict) in any family or series. This is used by commands \textunist, \textunibst, \unist, and \unibst.

The shape selected by \stshape.

These are the macros that switches to universal font, locally, i.e., for a few words, in various series and shapes. The names are pretty self explanatory, I think. These macros will always change the font series and shape according to it’s definition, nothing more, and nothing less, except \textuni, which only changes the current family to uni. Notice it simply uses the \unifamily command to do the trick. The second argument to \DeclareTextFontCommand should not normally contain commands that typeset, or commands not relevant to the selection of fonts, but to make up for the non-default characterers and character placements, we do include \unifamily in the argument.

Notice that only \textuni is defined if option strict was given to uni package.

\textunist
\textunirm
\textunibf
\textunisl
\textunibsl
\textunibsc

Local Font Selection

These are the macros that switches to universal font, locally, i.e., for a few words, in various series and shapes. The names are pretty self explanatory, I think. These macros will always change the font series and shape according to it’s definition, nothing more, and nothing less, except \textuni, which only changes the current family to uni. Notice it simply uses the \unifamily command to do the trick. The second argument to \DeclareTextFontCommand should not normally contain commands that typeset, or commands not relevant to the selection of fonts, but to make up for the non-default characterers and character placements, we do include \unifamily in the argument.

Notice that only \textuni is defined if option strict was given to uni package.

18 Change of \unishape on 98/08/01, Version 2.0
19 Change of \textuni on 98/08/01, Version 2.0
Global Font Selection

These macros work as \textuni... above, except the take effect from the point used, to the end of the current group. Again font selection is exactly as given in the command name, except for \uni which only changes the family.

Notice that only \uni is defined if option strict was given to uni package.

default Option Font  Now if you gave the default option to the package, \ifdef@ult evaluates to true, and so we setup the default font to be universal.

\textbf{Change of \uni on 98/08/01, Version 2.0}
in the medium or bold version, depending on whether you gave the medium option or not.

\ifdef@ult
\renewcommand{\familydefault}{uni}
\ifmedium\renewcommand{\seriesdefault}{m}\else\renewcommand{\seriesdefault}{b}\fi
\renewcommand{\itdefault}{sl}

We define \textcmr and \cmr are the logical extensions of \cmrfamily.

\DeclareRobustCommand{\cmrfamily}{%
\not@math@alphabet\cmrfamily\relax%
\fontencoding{OT1}\fontfamily{cmrdefault}\selectfont}
\newcommand{\cmrdefault}{cmr}
\newcommand{\cmrenc}{OT1}
\DeclareTextFontCommand{\textcmr}{\cmrfamily}
\DeclareOldFontCommand{\cmr}{\cmrfamily}{}
\fi

4.2 The Font Definition Files — *uni.fd

These files is needed in the New Font Selection Scheme, used by \TeX2ε, but really isn’t necessary for pure \TeX or \LaTeX2.09 users, but since those are treated alike, I did put in the extra effort and made the files.

What they really do, is to specify what font driver file should be loaded when the user switches font. Notice, a lot of the font shapes etc. available in the Computer Modern Roman scheme isn’t available in the universal font, so we substitute with whatever is closest, something from the universal or Computer Modern Roman scheme.

The reason why the file names may look a bit weird, is to make the font conform to the standard set by fontname used by kpathsea in most \TeX, \LaTeX, and METAFONT systems.

Math Encoding First comes the the font definition file for math, but since no math characters is defined in the universal font, we substitute with the relevant fonts from Computer Modern Roman.

\ProvidesFile{omluni.fd}
[1998/08/01 v2.0 Non Standard LaTeX font definitions]
\DeclareFontFamily{OML}{uni}{\skewchar\font127}
\DeclareFontShape{OML}{uni}{m}{n}{<-> ssub * cmm/m/it}{}
\DeclareFontShape{OML}{uni}{m}{it}{<-> ssub * cmm/m/it}{}
\DeclareFontShape{OML}{uni}{m}{sl}{<-> ssub * cmm/m/it}{}
\DeclareFontShape{OML}{uni}{m}{sc}{<-> ssub * cmm/m/it}{}
\DeclareFontShape{OML}{uni}{bx}{n}{<-> ssub * cmm/b/it}{}
\DeclareFontShape{OML}{uni}{bx}{it}{<-> ssub * cmm/b/it}{}
\DeclareFontShape{OML}{uni}{bx}{sl}{<-> ssub * cmm/b/it}{}
\DeclareFontShape{OML}{uni}{bx}{sc}{<-> ssub * cmm/b/it}{}
Symbols Encoding  Next is the definitions for symbols fonts, but as above, there is no separate symbol font defined for the universal font, so we substitute for default.

1 \ProvidesFile{omsuni.fd}
2 {1998/08/01 v2.0 Non Standard LaTeX font definitions}
3 \DeclareFontFamily{OMS}{uni}{
4 \skewchar\font 48 }
5 \DeclareFontShape{OMS}{uni}{m}{n}{<-> ssub * cmsy/m/n}{}
6 \DeclareFontShape{OMS}{uni}{m}{it}{<-> ssub * cmsy/m/n}{}
7 \DeclareFontShape{OMS}{uni}{m}{sl}{<-> ssub * cmsy/m/n}{}
8 \DeclareFontShape{OMS}{uni}{m}{sc}{<-> ssub * cmsy/m/n}{}
9 \DeclareFontShape{OMS}{uni}{bx}{n}{<-> ssub * cmsy/b/n}{}
10 \DeclareFontShape{OMS}{uni}{bx}{it}{<-> ssub * cmsy/b/n}{}
11 \DeclareFontShape{OMS}{uni}{bx}{sl}{<-> ssub * cmsy/b/n}{}
12

Normal Encoding  Now the definitions for the normal font. This is of course defined, anything else would be ludicruis. It is very straightforward most of the way. However, notice the substitutions at the end of the file.

1 \ProvidesFile{ot1uni.fd}\
2 {1998/08/01 v2.0 Non standard LaTeX font definitions}
3 \DeclareFontFamily{OT1}{uni}{\hyphenchar\font 45 }

Next comes the specifications of what to load in normal upright shape

5 \DeclareFontShape{OT1}{uni}{m}{n}{<5><6><7><8>fulmr8
6 <9>fulmr9
7 <10><10.95>fulmr10
8 <12><14.4>fulmr12
9 <17.28><20.74><24.88>fulmr17
10 }{}
11
12

Next comes the specifications of what to load in normal slanted shape

13 \DeclareFontShape{OT1}{uni}{m}{sl}{<5><6><7><8>fulmo8
14 <9>fulmo9
15 <10><10.95>fulmo10
16 <12><14.4>fulmo12
17 <17.28><20.74><24.88>fulmo17
18 }{}
19
20

Now for small caps medium definitions.

21 \DeclareFontShape{OT1}{uni}{m}{sc}{<5><6><7><8>fulmc8
22 <9>fulmc9
23 <10><10.95>fulmc10
24 <12><14.4>fulmc12
25 <17.28><20.74><24.88>fulmc17
26 }{}
27
Now for *strict medium* definitions.

\begin{verbatim}
\DeclareFontShape{OT1}{uni}{m}{st}{
<5><6><7><8>fulmst8
<9>fulmst9
<10><10.95>fulmst10
<12><14.4>fulmst12
<17.28><20.74><24.88>fulmst17
}\{}
\end{verbatim}

Next comes the specifications of what to load in **bold face upright shape**

\begin{verbatim}
\DeclareFontShape{OT1}{uni}{b}{n}{
<5><6><7><8>fulbr8
<9>fulbr9
<10><10.95>fulbr10
<12><14.4>fulbr12
<17.28><20.74><24.88>fulbr17
}\{}
\end{verbatim}

Next comes the specifications of what to load in **bold face slanted shape**

\begin{verbatim}
\DeclareFontShape{OT1}{uni}{b}{sl}{
<5><6><7><8>fulbo8
<9>fulbo9
<10><10.95>fulbo10
<12><14.4>fulbo12
<17.28><20.74><24.88>fulbo17
}\{}
\end{verbatim}

And **small caps bold face** definitions.

\begin{verbatim}
\DeclareFontShape{OT1}{uni}{b}{sc}{
<5><6><7><8>fulbc8
<9>fulbc9
<10><10.95>fulbc10
<12><14.4>fulbc12
<17.28><20.74><24.88>fulbc17
}\{}
\end{verbatim}

And **strict bold face** definitions.

\begin{verbatim}
\DeclareFontShape{OT1}{uni}{b}{st}{
<5><6><7><8>fulbst8
<9>fulbst9
<10><10.95>fulbst10
<12><14.4>fulbst12
<17.28><20.74><24.88>fulbst17
}\{}
\end{verbatim}

Now for the substitutions. This is straight forward, that is, upright slanted is substituted by upright; bold face condensed, and bold face extra is substituted by boldface; and italic by slanted\textsuperscript{21}.

\textsuperscript{21}As mentioned earlier, one of the characteristics of this font, is that it has no serifs, so it would be strange to include italics in this font.
T1 encoding  Since the T1 encoding relies on the font scheme of \texttt{exbase} (EC fonts), and \texttt{dxbase} (DC fonts), by Jörg Knappen and Nobert Schwartz, which is a whole new way of making fonts, that requires some special measures that I couldn’t incorporate into the \texttt{universal} font without a whole lot of work, I chose not to make support for the T1 scheme. Therefore this file issues a warning and then exists the whole \LaTeX{} run. This may seem a bit drastic, but it ensures that you don’t get screwed up results.

1 \ProvidesFile{t1uni.fd}
2 \[1998/08/01 v2.0 Non Standard \LaTeX{} font definitions\]
3 \typeout{WARNING: universal font not available in t1 definition.}
4 \typeout{WARNING: exiting, correct your source.}
5 \stop

U Encoding  This is similar to the normal encoding above, except encoding is U.

1 \ProvidesFile{uuni.fd}
2 \[1998/08/01 v2.0 Non Standard \LaTeX{} font definitions\]
3 \DeclareFontFamily{U}{uni}{\hyphenchar\font45}
4 \DeclareFontShape{U}{uni}{m}{n}{<5><6><7><8>fulmr8
5 <9>fulmr9
6 <10><10.95>fulmr10
7 <12><14.4>fulmr12
8 <17.28><20.74><24.88>fulmr17
9 }{}
10 \DeclareFontShape{U}{uni}{m}{s}{<5><6><7><8>fulmo8
11 <9>fulmo9
12 <10><10.95>fulmo10
13 <12><14.4>fulmo12
14 <17.28><20.74><24.88>fulmo17
15 }{}
16 \DeclareFontShape{U}{uni}{m}{sc}{<5><6><7><8>fulmc8
17 <9>fulmc9
18 <10><10.95>fulmc10
19 <12><14.4>fulmc12
20 <17.28><20.74><24.88>fulmc17
21 }{}}
A Solution to the \texttt{bauhausforms} problem

A.1 The Problem

This I owe to Joseph Collins \texttt{<collins@ARL.MIL>}. During the preperation of version 1.0 of this font, I was loosing sleep over what I chose to call ‘The \texttt{bauhausforms} problem’, after the problems the symbol \texttt{\textdegree} gave me.

As can be seen from the symbol, the idea is to make a figure out of a circle, a square, and a triangle. From these three figures you can ofcourse make infinitly many figures, even though the sizes are limited. The particular combination of the three figures I was looking, first seemed simple\textsuperscript{29}, but turned out to be difficult. What I wanted to do was (see also figure 1):

Take a circle of some radius (e.g., 1). Inside this circle draw a isosceles triangle $\triangle ABC$, with all vertices on the circle, The singular vertice $A$ placed on the horizontal line traveling left from the center of the circle.

Now draw a square $\square PQRS$ inside of the circle, having two vertices on the circle, and two on the triangle.

The wanted triangle and square are such that, the opposing side of the singular vertice $BC$ should divede the square into to equal oblongs, i.e., intersect $PQ$ midway between $P$ and $Q$.

This shouldn’t be to difficult, should it. Well I didn’t think so, but after many late evenings with pen, paper, ruler, compasses, and heavy use of trigonometric relations, I found out that the half $\varphi$ of the singular vertice should obey:

$$0 = \sin\left(\cos^{-1}\left(\frac{\cos 2\varphi}{\sqrt{2}}\right)\right) - \frac{\cos 2\varphi}{\sqrt{2}} - 2\sqrt{2} \frac{\cos^2\varphi \sin \varphi}{\cos \varphi + \sin \varphi}$$

\textsuperscript{29}And after having seen Mr. Collins solution, it did again.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure1.png}
\caption{The \texttt{bauforms} problem.}
\end{figure}
Now I dare you to find the exact solution to that.
Using numerical methods (Newton’s method), was of course no problem, and
gave satisfactory result. If the expression on the right above is labelled$f$,
$f’$ is:

\[
f’ = \frac{\sin 4\varphi - \sqrt{2} \sin 2\varphi}{2} \left[ \frac{3 - \cos 4\varphi}{4} \cos \varphi + \sin \varphi \right] - \frac{2\sqrt{2} \cos \varphi}{\cos \varphi + \sin \varphi} \left( (\cos \varphi + \sin \varphi) (\cos^2 \varphi - 2 \sin^2 2\varphi) \right) - \left( \cos \varphi - \sin \varphi \right) \cos^2 \varphi \sin \varphi.
\]

Using these expressions for $f$ and $f’$ in a Fortran program, I reached results close
to what Mr. Collins found.

### A.2 Joseph Collins’ Solution

In Mr. Collins notation, the points on figure 1 has the following coordinates:

- $A = (0, -1)$
- $B’ = (x, 0)$
- $C = (x, y)$
- $O = (0, 0)$
- $P’ = (x - h, 0)$
- $Q’ = (x + h, 0)$
- $R = (x + h, h)$
- $S = (x - h, h)$

Below is what Mr. Collins wrote me — thank you very much.

On the unit circle

\[x^2 + y^2 = 1\] (1)

we have the vertices of a triangle at $(-1, 0), (x, y),$ and $(x, -y)$. A square has four
vertices $(x \pm h, \pm h)$, where the two points $(x - h, \pm h)$ lie on the triangle (constraint A)
and the two points $(x + h, \pm h)$ lie on the circle (constraint B). Thus, the vertical
side of the triangle bisects the square. From constraint A, upon consideration of
similar triangles, we have

\[
\frac{y}{1 + x} = \frac{h}{1 + x - h}, \text{ so that } h = \frac{(1 + x)y}{1 + x + y}.
\]

By (1), this is

\[h = \frac{(1 + x)\sqrt{1 - x^2}}{1 + x + \sqrt{1 - x^2}}.\] (2)

From constraint B, we get

\[(x + h)^2 + h^2 = 1.\] (3)

Any simultaneous solution of (2) and (3) is also a solution of

\[8x^3 - 4x^2 - 3x + 1 = 0,\] (4)

the relevant solution being

\[x = \frac{1}{6} + \frac{\sqrt{11}}{18} \sin \left[ \frac{\pi}{6} - \frac{1}{3} \arctan \left( \frac{3\sqrt{37}}{23} \right) \right].\] (5)
Equation (4) and solution (5) courtesy of Mathematica. We get \( y \) and \( h \) from (1) and (2), respectively. The angle at \((-1,0)\) is \( \varphi = 2 \arctan(y/(1 + x)) \).

Approximate values are

\[
\begin{align*}
x & \approx 0.2865914 \\
y & \approx 0.9580529 \\
h & \approx 0.5491394 \\
\varphi & \approx 1.280129 \left( \approx 73.346^\circ \right)
\end{align*}
\]

\section*{B Copyrights — GNU General Public Licence}

This \texttt{METAFONT} implementation of the “universal” font and the \texttt{\LaTeX} support package “uni” copyright © 1998 Christian Holm.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but \textbf{\textit{without any warranty}}; without even the implied warranty of \textit{merchantability} or \textit{fitness for a particular purpose}. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to

The Free Software Foundation, Inc.
675 Mass Ave
Cambridge
MA 02139
USA

See the file \texttt{copyright} in the distribution for the complete GNU General Public License.

You can reach the me (the copyright holder) at

Christian Holm
Sankt Hansgade 23, 1. th.
DK–2200 Copenhagen N
Denmark
E-mail: cholm@nbi.dk or cholm@fys.ku.dk

\section*{C Wishlist}

Below is a list of things I would like to do with the font and package. If anyone has any suggestions, ready-made code, or new ideas, please let me know.

If you would like to take on one or more of the tasks presented below, please do so, but send me a note so that I may coordinate with my own efforts, and perhaps have a constructive discourse.

I should however instruct you to read the \textit{complete} documentation of the package and font, since this may give some reasons why I have chosen a particular approach.
• Make the font an 8-bit encoded (256 characters) font, conforming somewhat to the T1 encoding of the Cork fonts.

• Making a package (perhaps \texttt{mfdoc}), to make documentation of \texttt{METAFONT} sources, just like \texttt{doc} is fore \texttt{LaTeX} sources. This is ofcourse a independent project, and I will properly not work on it.
D  Font Charts

Below are some charts of the universal font in different series and shapes (medium upright, bold upright, medium slanted, medium small caps), all in size 8pt.
<table>
<thead>
<tr>
<th></th>
<th>'0'</th>
<th>'1'</th>
<th>'2'</th>
<th>'3'</th>
<th>'4'</th>
<th>'5'</th>
<th>'6'</th>
<th>'7'</th>
</tr>
</thead>
<tbody>
<tr>
<td>'0x'</td>
<td>00x</td>
<td>01x</td>
<td>02x</td>
<td>03x</td>
<td>04x</td>
<td>05x</td>
<td>06x</td>
<td>07x</td>
</tr>
<tr>
<td>'1x'</td>
<td>10x</td>
<td>11x</td>
<td>12x</td>
<td>13x</td>
<td>14x</td>
<td>15x</td>
<td>16x</td>
<td>17x</td>
</tr>
<tr>
<td>'2x'</td>
<td>20x</td>
<td>21x</td>
<td>22x</td>
<td>23x</td>
<td>24x</td>
<td>25x</td>
<td>26x</td>
<td>27x</td>
</tr>
<tr>
<td>'3x'</td>
<td>30x</td>
<td>31x</td>
<td>32x</td>
<td>33x</td>
<td>34x</td>
<td>35x</td>
<td>36x</td>
<td>37x</td>
</tr>
<tr>
<td>'4x'</td>
<td>40x</td>
<td>41x</td>
<td>42x</td>
<td>43x</td>
<td>44x</td>
<td>45x</td>
<td>46x</td>
<td>47x</td>
</tr>
<tr>
<td>'5x'</td>
<td>50x</td>
<td>51x</td>
<td>52x</td>
<td>53x</td>
<td>54x</td>
<td>55x</td>
<td>56x</td>
<td>57x</td>
</tr>
<tr>
<td>'6x'</td>
<td>60x</td>
<td>61x</td>
<td>62x</td>
<td>63x</td>
<td>64x</td>
<td>65x</td>
<td>66x</td>
<td>67x</td>
</tr>
<tr>
<td>'7x'</td>
<td>70x</td>
<td>71x</td>
<td>72x</td>
<td>73x</td>
<td>74x</td>
<td>75x</td>
<td>76x</td>
<td>77x</td>
</tr>
</tbody>
</table>

80
<table>
<thead>
<tr>
<th>'0'</th>
<th>'1'</th>
<th>'2'</th>
<th>'3'</th>
<th>'4'</th>
<th>'5'</th>
<th>'6'</th>
<th>'7'</th>
</tr>
</thead>
<tbody>
<tr>
<td>00x</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>00x</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>00x</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>00x</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>00x</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>00x</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>00x</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>00x</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

81