Installing Expert Fonts: Minion Pro

Abstract
Installing fonts for ConTEXt can be intimidating business. In this issue we take on a real monster: a collection of Adobe Minion Pro expert fonts. We hope our installation of this collection will provide an illustrative example for ConTEXt users, and help to ease the pain of installing new fonts (if you can install Minion Pro, Myriad Pro and Poetica, you can install just about anything!).

Introduction

Fonts can be a messy business in TEX (and, by extension, ConTEXt), and it's easy to get intimidated. One reason for this is TEX's flexibility; TEX allows you to create very sophisticated ways to take advantage of a font and to create, from one or more given font families, typeface collections tailored to your needs. Another reason is a (hopefully temporary) lack of standardization of map and encoding files between pdfe-TEX, dvips, and dvipdfmx. This second reason is not really a ConTEXt problem per se, though it certainly affects getting fonts working in ConTEXt.
Furthermore, ConTEXt handles fonts and font families by means of *typescripts*; these can be a bit disorienting to someone coming from LaTEX and the New Font Selection Scheme (NFSS). On the other hand, after initial hesitation (having myself migrated from the LaTEX world), I have concluded that the typescript approach is much more powerful and transparent than NFSS.
For a present book project, I decided to use a very complicated set of fonts from Adobe: Minion Pro (roman or serif), Myriad Pro (sans serif) and Poetica (calligraphy); all by Robert Slimbach. This set also includes a number of expert fonts with non-standard encodings. Together – and aside from mathematics – this set can provide a very nice alternative to the Computer/Latin Modern family, and one particularly suited for the humanities. These fonts also provide some of the few really excellent examples of *multiple master* (MM) technology, by Adobe. The promise of MM font technology was to provide a means of creating a series of finely optically scaled styles and alternative of a font from a single font file.[1]
On the other hand, despite its promise the system was never widely used and Adobe apparently no longer fully supports it.
In the present experiment we will focus on installing Minion Pro. I will not attempt to fine tune the weights; I will just use the defaults (mostly 2 weights per variation, plus a semibold style).[2] There is also a Minion Pro Opticals family, which I received while writing this issue. Although this tutorial is based on the older Minion Pro familiar to advanced LaTEX users, Appendix 1 explains how to set up Minion Pro Opticals. It should be easy to follow for anyone who has read the earlier sections, and provides a nice example of a truly advanced typescript.

Our work may be divided into three parts:

1. preparing the raw fonts;
2. installing the fonts; and
3. configuring typescripts and map files to use the fonts.

Ok, let's get to work!

Preparing the Fonts

Fonts generally will come in one of three forms: Type1 (`*.pfb`), TrueType (`*.ttf`), and OpenType (`*.otf`). TEX was generally restricted to Type1 fonts until recently. pdfe-TEX supports the other two to some degree. dvipdfmx supports large Type1 files (>256 characters per font); I don't know the status of its present or planned support for the other two.
Some fonts (like standard Type1 fonts) contain only a standard palette of 256 character-slots. In general, such fonts do not contain expert characters or glyphs such as 'ff', 'ffi', and 'ffl'. Given a standard font, we need to combine information from at least one other corresponding font to get a complete and professional typeface for that standard font. There are three ways to prepare the raw fonts for installation. One may use:

1. the fontinst package (for Type1 `*.pfb's`);
2. FontForge (formerly PfaEdit) (for Type1 `*.pfb's`); and
3. pre-prepared fonts, with standard, expert, and variant glyphs all in one font (TrueType and, more and more, OpenType).

If your fonts are already in a pre-prepared format, then you may just skim the first two subsections below.

fontinst

ConTEXt has its own font installation script, texfont. From page 1 of the texfont manual (`mtexfont.pdf`):

> The script only covers 'normal' fonts... Special fonts, like expert fonts, assume a more in depth knowledge of font handling. We may deal with them in the future. The more demanding user can of course fall back on more complicated tools like fontinst.

Although written in PlainTEX, the interface to fontinst is somewhat LaTEX-oriented. So its syntax largely follows the NFSS. This is no problem for ConTEXt: we only need the virtual fonts and tfm's produced by fontinst, and we ignore the `*.fd` file. Below we outline the procedure for preparing the fonts for installation using fontinst.[3]
Assuming that you are starting with 256-character Type1 fonts, you may rename them according to the older Berry convention.[4] We don't need that convention with today's operating systems but we will use it as a starting point. This is since LaTEX already has a setup for Minion Pro that uses the Berry fontname scheme and some readers may already have the raw fonts in this format.
The Minion Pro that I have contains 31 fonts. Here is a descriptive listing of the Type1 Minion Pro family:

```
Minion (31 fonts):

pmnb7d.pfb      Minion Bold Oldstyle Figures
pmnb8a.pfb      Minion Bold
pmnb8x.pfb      Minion Bold Expert

pmnbi7d.pfb     Minion Bold Italic Oldstyle Figures
pmnbi8a.pfb     Minion Bold Italic
pmnbi8x.pfb     Minion Bold Italic Expert
```

`pmnc7d.pfb`	`Minion Black Oldstyle Figures`
`pmnc8a.pfb`	`Minion Black`
`pmnc8x.pfb`	`Minion Black Expert`
`pmnr8a.pfb`	`Minion Regular`
`pmnr8x.pfb`	`Minion Regular Expert`
`pmnrc8a.pfb`	`Minion Regular Small Caps & Oldstyle Figures`
`pmnrd8a.pfb`	`Minion Regular Display`
`pmnrd8x.pfb`	`Minion Regular Display Expert`
`pmnrdc8a.pfb`	`Minion Regular Display Small Caps & Oldstyle Figures`
`pmnrdi8a.pfb`	`Minion Italic Display`
`pmnrdi8x.pfb`	`Minion Italic Display Expert`
`pmnrdic8a.pfb`	`Minion Italic Display Small Caps & Oldstyle Figures`
`pmnrdiw8a.pfb`	`Minion Italic Display Swash`
`pmnri8a.pfb`	`Minion Italic`
`pmnri8x.pfb`	`Minion Italic Expert`
`pmnric8a.pfb`	`Minion Italic Small Caps & Oldstyle Figures`
`pmnriw8a.pfb`	`Minion Italic Swash`
`pmnrp8a.pfb`	`Minion Ornaments`
`pmns8a.pfb`	`Minion Semibold`
`pmns8x.pfb`	`Minion Semibold Expert`
`pmnsc8a.pfb`	`Minion Semibold Small Caps & Oldstyle Figures`
`pmnsi8a.pfb`	`Minion Semibold Italic`
`pmnsi8x.pfb`	`Minion Semibold Italic Expert`
`pmnsic8a.pfb`	`Minion Semibold Italic Small Caps & Oldstyle Figures`
`pmnsiw8a.pfb`	`Minion Semibold Italic Swash`

Let us begin our analysis of Minion; we need to make a few decisions. Just make a note of them for later; it helps to stay organized with all the accounting involved in the typescripts:
We first note that, aside from the ornamental font, there are 5 main style variations: medium, semibold, bold, black, and italic. Medium has a display version, italic has a display version, bold has an italic version, and semibold has an italic version, for a total of nine variations. We need to make some sense of this in terms of optical scaling. For our future typescript, we will initially group some of these as follows:

- For `\tf`, let's try medium for sizes $<$ 17.3pt, and medium display for sizes $\geqslant$ 17.3pt;
- For `\bf`, try bold for sizes $\geqslant$ 8pt, and black for sizes $\leqslant$ 8pt (there is no display size for bold). Similarly for `\bi`;
- For `\it`, we try italic for sizes $<$ 17.3pt, and italic display for sizes $\geqslant$ 17pt;[5]
- We will leave semibold as its own alternative, although I did once try treating semibold (`\sb`) as an option for small or caption-sizes ($\leqslant$ 8pt). I think this was a failure, but the reader should try it and judge for himself.

The rest of our choices will be analogous.

We also note the following, based on a direct examination of these fonts:

- Based on the above grouping, small caps will be available in both weights for `\tf` and for `\it`, but not for `\bf` (sigh) or `\bi`. Oddly, semibold has both a small caps variation and a small caps italic variation. According to Lehman (page 63), semibold is the actual default bold weight; maybe he's right.[6]
- For a given optical size (as tentatively defined above), old style figures are available in both the expert font and in the old style figures font;
- For some reason, Minion Bold Oldstyle Figures as well as Minion Bold Italic Oldstyle Figures have no small caps; each are identical to Minion Bold and Minion Bold Italic respectively, except for the numerals. The other styles use small caps in their old style figures versions;
- It is our intention to make old style numerals the default for our entire typeface collection; this makes sense in the humanities, I think;[7]
- For all five primary variations (`\tf`, `\it`, `\bf`, `\bi`, and `\sb`) and their derivatives, we will try
  — using the expert fonts for both old figures and expert ligatures;
  — using the old style figures fonts for small caps only (`\tf` and `\it`);
  Although we could just default to the old style figures fonts for `\bf` and `\bi`, for consistency purposes we will, for the time being, treat all four typefaces equal in this regard. You can always change this. . .
- The most difficult task to accomplish the above is dealing with the expert fonts in this collection. They share an non-standard encoding vector. We need to make our typeface collection default to the expert ligatures and to the old style numerals.

Preparing the raw fonts for installation involves making a fontinst file `makeminion.tex` like the following:

```
\input fontinst.sty

\installfamily{T1}{pmn}{}
\installfonts

% minionr
\installfont{minionr10}   {pmnr8a,pmnr8x,latin}     {T1j}{T1}{minion}{m}{n}{}
\installfont{minionr17}   {pmnrd8a,pmnrd8x,latin}   {T1j}{T1}{minion}{m}{n}{}

% minioni
\installfont{minioni10}   {pmnri8a,pmnri8x,latin}   {T1j}{T1}{minion}{m}{it}{}
\installfont{minioni17}   {pmnrdi8a,pmnrdi8x,latin} {T1j}{T1}{minion}{m}{it}{}

% minionb
\installfont{minionb10}   {pmnb8a,pmnb8x,latin}     {T1j}{T1}{minion}{b}{n}{}
\installfont{minionbl10}  {pmnc8a,pmnc8x,latin}     {T1j}{T1}{minion}{b}{n}{}

% minionbi
\installfont{minionbi10}  {pmnbi8a,pmnbi8x,latin}   {T1j}{T1}{minion}{m}{bi}{}

% minionsc
\installfont{minionsc10}  {pmnrc8a,latin}           {T1j}{T1}{minion}{m}{sc}{}
\installfont{minionsc17}  {pmnrdc8a,latin}          {T1j}{T1}{minion}{m}{sc}{}

% minionisci
\installfont{minionsci10} {pmnric8a,latin}          {T1j}{T1}{minion}{m}{sc}{}
\installfont{minionsci17} {pmnrdic8a,latin}         {T1j}{T1}{minion}{m}{sc}{}
```

```
% minionsb
\installfont{minionsb10}  {pmns8a,pmns8x,latin}    {T1j}{T1}{minion}{sb}{n}{}

% minionsbi
\installfont{minionsb10}  {pmnsi8a,pmnsi8x,latin}  {T1j}{T1}{minion}{sb}{it}{}

% minionsbsc
\installfont{minionsc8}   {pmnsc8a,latin}          {T1j}{T1}{minion}{sb}{sc}{}
\installfont{minionsci8}  {pmnsic8a,latin}         {T1j}{T1}{minion}{sb}{sc}{}

% minionisw
%\installfont{minionswi10}  {pmnriw8a,latin}         {T1j}{T1}{minion}{m}{it}{}
%\installfont{minionswi17}  {pmnrdiw8a,latin}        {T1j}{T1}{minion}{m}{it}{}
%\installfont{minionsbswi10}{pmnsiw8a,latin}         {T1j}{T1}{minion}{sb}{it}{}

% miniono
%\installfont{miniono10}   {pmnrp8a,latin}         {T1j}{T1}{minion}{m}{n}{}

\endinstallfonts

\bye
```

Let us look briefly at the first `\installfont` line (see Hoenig or Lehman for details):

- `\installfont {minionr10}`
  The name of our virtual font will be `minionr10`;
- `{pmnr8a,pmnr8x,latin}`
  Our standard font is `pmnr8a`, expert font is `pmnr8x`, and `latin.mtx` is the default fontinst metric file that defines at least 401 glyphs found in Latin alphabets (see Hoenig, page 180);
- `{T1j}{T1}{minion}{m}{n}{}`
  The *encoding file* is `t1j.etx` (Cork with oldstyle numerals), *general encoding* is T1 (cork), *family* is minion, *series* is medium, *shape* is normal, and size is left empty. This is all NFSS terminology.

Note that we have intentionally organized `makeminion.tex` to be analogous to our future typescript file. Also, we have commented out the swash and ornament lines. This is because I personally prefer to deal with the preparation, installation, and configuration of each of these two in its own directory, separate from the main fonts. So in the `/swash` subdirectory `makeminionsw.tex` will contain only the swash lines, and `/ornaments` will contain only the ornament line. Looking ahead, we will have three separate typescript classes: main, swash, and ornament. Because writing advanced typescripts requires a lot of careful accounting, it is better to keep these classes separate. If you don't believe me, try doing everything that follows in the configuration stage in a single typescript. You'll see;-)
In the LaTeX version, the final fonts are given names like `pmnr9e` instead of `minionr10`. Since we don't have to deal with NFSS and old encodings, we can happily dispense with that here.
Note: In retrospect, I prefer to avoid fontinst. There is a very limited number of pre-made `*.etx` files, though you can make your own. But if you have a set of 256-character-slots Type1 fonts, the next method will make our life a bit easier later, as you'll see. On the other hand, if you really need dvips, then you may need to go the fontinst route (dvipdfmx works fine with the next method).

FontForge
While I was messing with fontinst, the following thought occurred to me: is there some way we can merge the expert and standard fonts into a single font file, so we

can just use texfont (you will soon see why this makes things easier)? I tried FontLab: no such feature. Fontographer? Foiled again. Then I looked at the open source FontForge (formerly PfaEdit).[8] For Windows users, there is a version for Cygwin. It's definitely worth installing a minimal Cygwin to have; instructions are on the FontForge site.

In FontForge, open `pmnr8a.pfb`. Then go to `ELEMENT => MERGE` to choose the corresponding expert font, `pmnr8x.pfb`. FontForge will add every character with a different name to the original glyph palette. Then save this new font to `minionr10.pfb`. You must repeat this for all standard fonts that have an expert companion. You can use `makeminion.tex` in the above subsection on fontinst to identify the correspondences and correct names. For those fonts that have no expert companion, just copy and rename them to our scheme.

A nice thing about FontForge is that it is scriptable. So those who are familiar with that can write a script so that FontForge can do all of this in batch. That skill is a bit beyond me, so I just did it the point-and-click way. Look up "scripting" in the FontForge documentation.

Pre-prepared Fonts

If you have OpenType or TrueType versions of the fonts then you are set. If you need to use aleph (ℵ, which cannot use `*.ttf/*.otf files`); or need dvipdfmx, then all you need to do is convert each font to Type1.[9] Don't worry about the 256-character-slot limit for Type1 fonts; it won't affect things for us. To follow along easily, save copies of your OpenType Minion Pro fonts to the names we are using here.

Installing the Fonts

First, we must have the `afm` files for all raw fonts. You can generate them with any decent font-editing software. There is an afm-generation utility, `getafm` that comes with TEXLive, but it does not procure the proper kerning info. A package of metrics for the *Adobe Type Classics for Learning* suite (including Minion Pro) is available from `http://www.lcdf.org/type/`.

fontinst

http://www.ntg.nl/maps/33/minion/fontinst.bat is a batch file that handles most of the work. Once you have generated the files and directories, you can install them in your local tree or in `/texmf-fonts`, which ConTEXt uses. You will also need a proper map file, which is where I made my big mistake with fontinst. For dvips I used lines like this:

```
pmnr8a     pmnr8a     <pmnr8a
```

But I discovered that I needed lines like this:

```
pmnr8a Minion-Regular "TeXBase1Encoding ReEncodeFont" <8r.enc<pmnr8a.pfb
```

Walter Schmidt has provided a complete LaTEX package for Minion:
`http://www.ctan.org/tex-archive/fonts/psfonts/w-a-schmidt/pmn.zip`
For details see Walter's package. Between this and Tutorial VI of Lehman's *Guide* you will learn all you need to know about installing Minion Pro, as well as a lot about fontinst. In any case, I much prefer using texfont. Our installation in texfont will involve multiple encodings. To do this in fontinst you may have to write your own `*.etx` files, endure a lot of debugging, and so forth. Make your life easy and get FontForge:-)

TEXfont: Type1, TrueType or OpenType Big Fonts
Let us begin by making three temporary directories:

- `/main`
  Place all Minion `pfb`'s and `afm`'s here;
- `/swash`
  Move all three swash fonts, `minionswi10.pfb`, `minionsbswi10.pfb`, and `minionswi17.pfb` here;
- `/ornament`
  Move the ornament font `miniono10.pfb` and metric file here.

Here we set our encoding vectors for Minion Pro. We will use texnansi encoding as our base, though you can easily choose another (like ec) if you like. Actually, we will use the file `texnansi-lm.enc`, in /texmf-local/fonts/enc/dvips/lm, as our base file, because it is easier to edit than `texnansi.enc`. Just make sure to remove all `*.dup` extensions. For example, change `/OE.dup` to `/OE`.
Now we create a few encoding files (all go into `/main` except the last two). From careful study of these examples, you can easily make your own special encodings at will. **Note**: each encoding file must have precisely 256 character lines, not counting the beginning line and the ending line:

- `texnansi-axo.enc`
  The prefix `texnansi` is important; it tells us that texnansi encoding is our foundation. The string 'ax' stands for 'Adobe Expert'. Finally, 'o' stands for 'old style numerals'. This encoding file will be used to create and install a virtual font that defaults to old style numerals. Simply replace the lines

```
/zero
/one
/two
/three
/four
/five
/six
/seven
/eight
/nine
```

  with

```
/zerooldstyle      %/zero etc.
/oneoldstyle       %:
/twooldstyle
/threeoldstyle
/fouroldstyle
/fiveoldstyle
/sixoldstyle
/sevenoldstyle
/eightoldstyle
/nineoldstyle
```

  The comments just remind us of the original characters we are replacing. Change the beginning line to `/enctexnansiaxo[`. There a couple of minor quirks to keep in mind. See http://www.ntg.nl/maps/33/minion/texnansi-axo.enc for the full `texnansi-axo.enc`; changes from the original `texnansi.enc` are noted. For example, there is no dottless 'j' in either the Adobe standard or expert encodings, at least not with Minion Pro.[10]

- `texnansi-axu.enc`
  This encoding file will be used to create and install a virtual font that defaults to upright numerals. Use the default numeral characters from `texnansi.enc`;
- `texnansi-axs.enc`
  This encoding file will be used to create and install a virtual font that defaults to superior numerals. Use `/zerosuperior`, etc.;
- `texnansi-axi.enc`
  This encoding file will be used to create and install a virtual font that defaults to inferior numerals. Use `/zeroinferior`, etc..
- `texnansi-axuc.enc`
  This encoding file will be used to create and install a virtual font that defaults to upright numerals and small caps. The small caps fonts that come with Minion Pro all default to old style numerals, and these numerals are encoded with the upright character names. Take `texnansi-axu.enc` and replace

  ```
  /a
  /b
  /c
  ```

  with

  ```
  /Asmall
  /Bsmall
  /Csmall
  ```

  and so forth. Using this particular encoding is only good for those standard fonts with small caps in the corresponding expert font. For example, Minion Bold Expert has no small caps (although Semibold Expert does). Basically you will be replacing all of the original small caps fonts with non-small caps big fonts encoded with small caps glyphs. We will say more about this below, in the section on typescripting.
- `texnansi-ao.enc` and `texnansi-aw.enc`
  We make encodings for the swashes and ornaments. The ornamnents take up 23 slots corresponding to A through W; the swashes take up A through Z. The `/space` slot is the only other one kept in place; fill up the rest with `/.notdef`'s, e.g.,

  ```
  /.notdef
  /.notdef
  /.notdef
  /ornament1      %/A,
  /ornament2      %/B
  /ornament3      %/C, etc
  ```

  for ornaments, and

  ```
  /.notdef
  /.notdef
  /.notdef
  /A
  /B
  /C
  /D              % etc.
  ```

  for swashes.

There are lots of other possibilities, like an encoding that uses text-fractions and so forth. You are now in control!
It is now time to install. texfont will do most of the work, but you have to install the encoding files by hand. It would be nice if texfont could do this for us as well. In the meantime, copy the encoding files to `/texmf-fonts/fonts/enc/dvips/minion`.
Do Not forget to install the encoding files!
Now we are ready to install our main fonts with texfont. The directory `/main` should have the `*.pfb` files, the `*.afm` files, and the encoding files (or you can pre-install the encoding files and do TEXHASH). From each of the three respective directories, issue the corresponding commands from the following:

```
texfont --ma --in --en=texnansi-axo --ve=adobe --co=minion --show

texfont --ma --in --en=texnansi-axu --ve=adobe --co=minion --show

texfont --ma --in --en=texnansi-axs --ve=adobe --co=minion --show

texfont --ma --in --en=texnansi-axi --ve=adobe --co=minion --show

% texfont --ma --in --en=texnansi-axuc --ve=adobe --co=minion --show

% uncomment if small~caps with upright numerals are desired
```

Do `texfont --help` to see the meaning of each of the above switches. The above commands use abbreviated versions (first two letters) of these options.[11]
Now you will find four pdf files in `/main`: `texnansi-axo-adobe-minion.pdf`, `texnansi-axu-adobe-minion.pdf`, `texnansi-axs-adobe-minion.pdf`, and `texnansi-axi-adobe-minion.pdf`. Take a look at these; they include beautiful font charts of your encodings. Also take a look at the map files in `/texmf-fonts/fonts/map/pdftex/context`. Peruse especially the way the virtual fonts and `tfm` files fonts are named. It's verbose but very easy to read and systematic.
Did you remember to install the encoding files?

Configuration

The texnansi-axo Typface Collection

Now we need to generate a set of typescripts that can handle our main Minion font collection: let's call them `type-mino.tex`, `type-minu.tex`, `type-mins.tex`, and `type-mini.tex`. All four are almost identical so we will analyze one of them in detail, `type-mino`. Each typescript will have five main parts: *font mapping*, *general names*, *font sizes*, *map loading*, and *final typefaces*. Let us deal with each of these in turn.

- Font Mapping
  Here we map the raw fonts to easy-to-understand names. Note that we are mapping, not directly to the `pfb`'s, but to the virtual fonts.

```
% We need a few switches: I don't guarantee that
% they don't conflict with other commands;-)

\definestyle [italicsmallcaps,smallcapsitalic] [\si] []
\definestyle [black]                           [\bk] []
\definestyle [semiboldroman,semibold]          [\sb] []
\definestyle [semibolditalic]                  [\st] []
\definestyle [semiboldsmallcaps]               [\sp] []
\definestyle [semiboldsmallcapsitalic]         [\stp][]

% Regular serifs, greater than 8pt, less than 17.3pt
```

```
\starttypescript[serif]                    [miniono] [texnansi-axo]

\definefontsynonym [Minion10]              [texnansi-axo-minionr10]
\definefontsynonym [Minion17]              [texnansi-axo-minionr17]

\definefontsynonym [MinionItalic10]        [texnansi-axo-minioni10]
\definefontsynonym [MinionItalic17]        [texnansi-axo-minioni17]

\definefontsynonym [MinionBold10]          [texnansi-axo-minionb10]
\definefontsynonym [MinionBlack]           [texnansi-axo-minionbl10]

\definefontsynonym [MinionBoldItalic]      [texnansi-axo-minionbi10]

\definefontsynonym [MinionCaps10]          [texnansi-axu-minionsc10]
\definefontsynonym [MinionCaps17]          [texnansi-axu-minionsc17]

\definefontsynonym [MinionItalicCaps10]    [texnansi-axu-minionsci10]
\definefontsynonym [MinionItalicCaps17]    [texnansi-axu-minionsci17]

\definefontsynonym [MinionSemiBold]        [texnansi-axo-minionsb10]

\definefontsynonym [MinionSemiBoldItalic]  [texnansi-axo-minionsbi10]

\definefontsynonym [MinionSemiBoldCaps]    [texnansi-axu-minionsbsc10]

\definefontsynonym [MinionSemiBoldItalicCaps][texnansi-axu-minionsbsci10]

\stoptypescript
```

Note that we map to the upright-encoded fonts for the six fonts with small caps. This is because the small caps fonts each defaults to old style numerals, but those numerals are encoded in the font with upright names. Furthermore, the small caps fonts do not have corresponding experts. So the small caps virtual fonts in `texnansi-axo` encoding have no numerals at all. On the other hand, the `texnansi-axu` encoded small caps virtual fonts will display old style numerals because those numerals are encoded in the font with upright names. The rest will display upright numerals. This inconsistency is wholly due to the manufacturer of the original raw fonts.
Similarly, the typescript for `texnansi-axu` encoded fonts will need to map small caps to the `texnansi-axuc` encoded fonts, if full consistency is desired. The `texnansi-axuc` encoded fonts do not need their own typescript, since they are just meant to supplement texnansi-axu.[12]
Note the option `[miniono]`. For `type-minu.tex` it should be `[minionu]` (with a 'u') and so forth.

- General Names
  This part may seem redundant right now, but it will make sense when we add the Myriad Pro collection. That is a sans serif, while Minion is a serif, so this helps keep things clear and organized.

```
\starttypescript[serif]                 [miniono] [name]

\definefontsynonym [Serif]              [Minion10]
\definefontsynonym [Serif17]            [Minion17]

\definefontsynonym [SerifItalic10]      [MinionItalic10]
\definefontsynonym [SerifItalic17]      [MinionItalic17]

\definefontsynonym [SerifBold10]        [MinionBold10]
\definefontsynonym [SerifBlack]         [MinionBlack]

\definefontsynonym [SerifBoldItalic]    [MinionBoldItalic]
```

```
\definefontsynonym [SerifCaps10]           [MinionCaps10]
\definefontsynonym [SerifCaps17]           [MinionCaps17]

\definefontsynonym [SerifItalicCaps10]     [MinionItalicCaps10]
\definefontsynonym [SerifItalicCaps17]     [MinionItalicCaps17]

\definefontsynonym [SerifSemiBold]         [MinionSemiBold]

\definefontsynonym [SerifSemiBoldItalic]   [MinionSemiBoldItalic]

\definefontsynonym [SerifSemiBoldCaps]     [MinionSemiBoldCaps]

\definefontsynonym [SerifSemiBoldItalicCaps][MinionSemiBoldItalicCaps]

\stoptypescript
```

- Font Sizes
  This is where we implement optical scaling (what little there is, anyway). If you have Minion Pro Opticals, you will have more choices. The following typescript will give you the needed insight to implement your own scheme for optical scaling.
  Note that in the first line of this section of our typescript, we have mapped `Minion10` to, not `Serif10`, but to just `Serif`. ConTEXt treats the `Serif` font as the default or empty font; if it is not defined, in a few cases ConTEXt will fall back to a typeface where it is defined (generally **Latin Modern**).[13]

```
\starttypescript [serif] [miniono] [size]

\definebodyfont [9pt,10pt,11pt,12pt,14.4pt]
  [rm]
  [tf=Serif sa 1,
   sc=SerifCaps10 sa 1,
   it=SerifItalic10 sa 1,
   si=SerifItalicCaps10 sa 1]

\definebodyfont [4pt,5pt,6pt,7pt,8pt]
  [rm]
  [tf=Serif sa 1,
   sc=SerifCaps10 sa 1,
   it=SerifItalic10 sa 1,
   si=SerifItalicCaps10 sa 1,
   bf=SerifBlack sa 1]

\definebodyfont [17.3pt,20.7pt,24.9pt]
  [rm]
  [tf=Serif17 sa 1,
   sc=SerifCaps17 sa 1,
   it=SerifItalic17 sa 1,
   si=SerifItalicCaps17 sa 1]

\definebodyfont [9pt,10pt,11pt,12pt,14.4pt,17.3pt,20.7pt,24.9pt]
  [rm]
  [bf=SerifBold10 sa 1]

\definebodyfont
  [24.9pt,20.7pt,17.3pt,14.4pt,12pt,11pt,10pt,9pt,8pt,7pt,6pt,5pt,4pt]
  [rm]
  [bi=SerifBoldItalic sa 1,
   sb=SerifSemiBold sa 1,
   st=SerifSemiBoldItalic sa 1,
   sp=SerifSemiBoldCaps sa 1,
   stp=SerifSemiBoldItalicCaps sa 1]

\stoptypescript
```

Note: Some of these switches are newly defined (like `\sp`), and the ConTEXt mechanism for enlarging and reducing the size of a given style variation will not work. We need to define them for completeness. See pages 129–131 of *ConTEXt: the Manual* for details. It's really quite straightforward, just a bit tedious and verbose, so we leave it as an exercise for the reader.[14] See also the typescript in http://www.ntg.nl/maps/33/minion/type-mpoo.tex. Choosing optical sizes is an area that needs a bit of experimenting to get exactly right. For example, does the black font really work at small bold sizes?

- Map Loading
  Here we load our map files, created during installation.

```
\starttypescript[map] [miniono] [texnansi-axo]

\loadmapfile[texnansi-axo-adobe-minion.map]
\loadmapfile[texnansi-axu-adobe-minion.map]

\stoptypescript
```

Here we also need the `texnansi-axu` map for the small caps as discussed above. Note that `type-minu.tex` will also need to load the `texnansi-axuc` map file, if you have installed and desire to have small caps with upright numerals.[15]

- Final Typefaces
  This is where we put it all together, our Minion typeface collection. We also define those fonts that do not come with Minion, Myriad, or Poetica, such as math fonts (we use Euler) and monospaced (we use Latin Modern). Note the 'o' suffix in what follows. Such identifying suffixes will be needed in the other typescript files as well as well.
  Note: While very powerful and transparent, typescripts are quite sensitive to these kinds of seemingly minor accounting issues, so be careful.

```
\starttypescript[ADOBEMiniono]

\definebodyfontenvironment
  [adobeminiono]
  [default]
  [interlinespace=2.6ex]

\definetypeface [adobeminiono]
[rm] [serif] [miniono] [miniono] [encoding=texnansi-axo]

% Myriad and Poetica to be configured later, then uncomment
%\definetypeface [adobeminiono]
%[ss] [sans] [myriado] [myriado] [encoding=texnansi]

%\definetypeface [adobeminiono]
%[cg] [calligraphy] [poetica] [poetica] [encoding=texnansi]

\definetypeface [adobeminiono]
[mm] [math] [euler] [default] [encoding=texnansi,rscale=0.89]

\definetypeface [adobeminiono]
[tt] [mono] [modern] [default] [encoding=texnansi,rscale=0.99]

\stoptypescript
```

We note that the non-Minion fonts used in our typeface collection, such as Euler math fonts and Latin Modern monospaced, need to be scaled. That is what the `rscale=<scale factor>` option does for us. We also note that Minion needs a smaller interline space factor than the usual 2.8`ex`. We may need to do some more testing in this regard, though 2.6`ex` seems to work well for the Minion design.
Note: Be aware that ConTEXt sets up `\em` with the slanted (`\sl`) style variation by default. But Minion Pro does not come with a slanted font. So `\em` will not work unless you map one of your fonts – see the previous section on size definitions – to `\sl`. Declare

```
\setupbodyfontenvironment[default][em=italic]
```

either in your typescript or in your style/environment file. It may not be such a good idea to define it in the typescript, because you could get odd results depending on the order your typescripts are scanned during compilation (assuming you've setup `\em` differently somewhere else).

Now write the above set of typescripts to a file, `type-mino.tex`. We can now test our typescript so far. Here is a test file:

```
% output=pdf interface=en

\usetypescriptfile[type-mino]
\usetypescript[ADOBEMiniono]
\switchtotypeface[adobeminiono]%

\starttext

This is a test of Minion in \CONTEXT. 1234

\bf This is a test of Minion in \CONTEXT. 1234

\it This is a test of Minion in \CONTEXT. 1234

\bi This is a test of Minion in \CONTEXT. 1234

\sc This is a test of Minion in \CONTEXT. 1234

\si This is a test of Minion in \CONTEXT. 1234

\sb This is a test of Minion in \CONTEXT. 1234

\stp This is a test of Minion in \CONTEXT. 1234

\switchtotypeface[adobeminionor]
{\tf ABCDEFGHIJKLMNOPQRSTUVW \par}\blank

\switchtotypeface[adobeminionsw]
{\sw ABCDEFGHIJKLMNOPQRSTUVW \par}\blank

\stoptext
```

Compiling gives us:

This is a test of Minion in ConTEXt. 1234

This is a test of Minion in ConTEXt. **1234**

This is a test of Minion in ConTEXt. *1234*

This is a test of Minion in ConTEXt. ***1234***

THIS IS A TEST OF MINION IN ConTEXt. 1234

THIS IS A TEST OF MINION IN ConTEXt. *1234*

This is a test of Minion in ConTEXt. **1234**

THIS IS A TEST OF MINION IN ConTEXt. ***1234***

ABCDEFGHIJKLMNOPQRSTUVW

Sample application

We end with an application: In our Minion installation and configuration we have a superior numerals typeface. This looks better than either upright or old style numerals for footnote marking. This example compares superior and upright numerals in footnotes, with old style numerals in the running text:

Consider what is said, not who has said it.[1234]

[1234] This aphorism is by ^{c}Alī ibn Abī Ṭālib (d. 661CE/AH).

Intermezzo 1 Upright footnote numerals.

Consider what is said, not who has said it.[1234]

[1234] This aphorism is by ^{c}Alī ibn Abī Ṭālib (d. 661CE/AH).

Intermezzo 2 Superior footnote numerals.

Post-dvi processing

Unfortunately, inconsistencies between pdfe-TEX, dvips, and dvipdfmx mean we have to do more work if we need post-`dvi` processing for any reason (this is the case with ℵ, for example).

dvipdfmx

You will need to

- write at least one map file for your collection;
- Look at, e.g., `texnansi-axo-adobe-minion.map`. Change the syntax from

```
texnansi-axo-raw-minionr10 Minion-Regular 4 < minionr10.pfb texnansi-axo.enc
```

to

```
texnansi-axo-raw-minionr10 texnansi-axo minionr10
```

- Now make your map file available to dvipdfmx. You can add a line like

  ```
  f minion-dvipdf.map
  ```

  to the file `/texmf-local/fonts/dvipdfm/config/config`, or you may call your map file from the command line

  ```
  $ dvipdfmx -f minion-dvipdf.map
  ```

dvips
If you need to use dvips, you may have to go the fontinst route. This is because dvips apparently looks for Type1 fonts with a 256-glyph limit, and ours (as well as **Latin Modern**) are bigger. This limitation seems a bit outdated, and hopefully the maintainers of dvips will one day remove this limitation. In any case, see Tutorial VII of Lehman's *Guide* for a very thorough discussion of preparing map files for dvips with fontinst.

I hope that you have found this article clear and enjoyable. See also *This Way* # 9, Using Platform Fonts, by Hans Hagen, for more on font installation.[16]
Best wishes for painless font installation in ConTEXt![17]

Minion Pro Opticals

As I was finishing this issue I received the complete Minion Pro Opticals set, in OpenType format. This set is more internally coherent than the older version we used for this tutorial. It contains six style variations: medium or regular, semibold, bold, italic, semibold italic, and bold italic. The black style variation is apparently gone. Each style variation comes in four optical sizes: normal, caption, subhead, and display. Each font has a standard 256-character encoding, plus a set of old style numerals, superiors, inferiors, a set of small caps (we have bold small caps now!), ornaments and hundreds of other alternates. There is no dedicated small caps or ornaments font. Each italic font has a large palette of swashes, many more than the original swash fonts. There is also Greek, Cyrillic, and lots of alternate or fancy ligatures. This is really much better than the original set.
Our encodings prepared earlier will suffice with a few changes (and you can always make your own):

- `texnansi-axo.enc`, `texnansi-axu.enc`, `texnansi-axs.enc`, and `texnansi-axi.enc` will stay the same;
- For small caps we now need a `texnansi-axoc.enc` for small caps with old style numerals. Just modify `texnansi-axuc.enc` and replace the default numerals with the old style ones;
- `texnansi-aw.enc` will have to change `/Aswash` to `/A.swash`, etc.. The italics font also offer lots of swash capitals with accents. Some of these swashes do not have a corresponding entry in the standard encoding: for example, there is not `/Ebreve` in the standard encoding to match `/Ebreve.swash`. So if you want the esoteric swashes you will have to pick and choose how you want to encode this within a 256-character context.

One idea about swashes: since they are now part of the full italics fonts, treat them like small caps, and encode them in the same `/a--/z` band of the encoding. This was much less trivial to accomplish in the old fonts.
- The names of the ornaments in `texnansi-ao.enc` will have to be changed: `/ornament1` becomes `/orn.001`, etc., up to `/orn.023`. An identical set of ornaments is present in every font, so you can also encode ornaments like a small caps font if you like.

Installation is just as before. Convert fonts to `*.pfb` (with `*.afm`), place in a separate directory with your encodings, then run texfont for as many encodings as you like.[18]
The typescript files are mostly as before: the only really interesting difference is the much better optical scaling. According to the Minion Pro Opticals documentation, the intended optical scaling spectrum is as follows:

- Caption: 6–8.4 point
- Normal (Regular): 8.5–13 point
- Subhead: 13.1–19.9 point
- Display: 20+ point

For small caps, one may choose to write a separate typescript file and typeface collection, in which case one has to switch fonts to use small caps. Or one can integrate, e.g, the small caps fonts into the upright numerals typescript file. Experiment to get the combination that works best for you.
We give the typescript files different names from before: `type-mpoo` for Minion Pro Opticals old style, `type-mpou` for upright, and so forth.
One full possible typescript, that for `type-mpoo`, including small caps, is available from http://www.ntg.nl/maps/33/minion/type-mpoo.tex. It should reward careful study.
Enjoy!

Notes

1. For details, see "Designing Multiple Master Typefaces," by Adobe: `http://partners.adobe.com/public/developer/en/font/5091.Design_MM_Fonts.pdf`.
2. We use the expressions 'style', 'variation', and 'family' in the senses employed in *ConTEXt: the Manual*, page 91. Adobe Minion Pro is a font *family* or typeface *family*, roman and sans serif are *styles*, bold and italic are *style variations*. In the ConTEXt world, the expression 'typeface' is often used to mention a user-defined collection of fonts, often drawn from various families.
3. For a wealth of details about fontinst and virtual fonts, see Alan Hoenig's book *TEX Unbound*. A more recent and up-to-date manual is *The Font Installation Guide*, by Phillip Lehman. It is available in CTAN:/info/type1fonts/fontinstallationguide.
4. For details, see Hoenig, pages 132–134, and Lehman, pages 11–13.
5. This is all intentionally experimental. Lehman, page 63 has more professional suggestions, but I think it's important to reflect ourselves. Probably you will one day have to install a font where no one has made predeterminations about this sort of thing.
6. On the other hand, Minion Pro Opticals has small caps for bold, and the official documentation seems to indicate that the default bold is, indeed, Minion Bold.
7. In *The Elements of Typographic Style*, Bringhurst enjoins:

> *Use titling [upright] figures with full caps, and text [old style] figures in all other circumstances.*

8. Available here: `http://fontforge.sourceforge.net/` .
9. One may use FontForge for this. There is also a tool `cfftot1` provided by lcdf (`http://www.lcdf.org/type/`) but it can not, as far as I can tell, generate an `*.afm` file. But see Appendix 1.
10. There is a free tool, `t1dotlessj`, that creates a dotless-'j' Type1 font from an existing standard font:

http://www.lcdf.org/type/t1dotlessj.1.html .
You may then use FontForge to merge this with your main font (preferable), or go through fontinst.
11. Adam Lindsay pointed out to me that you may also use the `--variant` option (e.g., `--va=texnansi-axo` instead of `--en`. Then there may be no need to install the encoding files: just use `[encoding=texnansi]` in the typescripts.
12. In `type-minu.tex`, replace the raw font names in the following lines

```
\definefontsynonym [MinionCaps10]             [texnansi-axu-minionsc10]
\definefontsynonym [MinionCaps17]             [texnansi-axu-minionsc17]

\definefontsynonym [MinionItalicCaps10]       [texnansi-axu-minionsci10]
\definefontsynonym [MinionItalicCaps17]       [texnansi-axu-minionsci17]

\definefontsynonym [MinionSemiBoldCaps]       [texnansi-axu-minionsbsc10]

\definefontsynonym [MinionSemiBoldItalicCaps] [texnansi-axu-minionsbsci10]
```

with the corresponding names from the `texnansi-axuc`:

```
\definefontsynonym [MinionCaps10]             [texnansi-axuc-minionr10]
\definefontsynonym [MinionCaps17]             [texnansi-axuc-minionr17]

\definefontsynonym [MinionItalicCaps10]       [texnansi-axuc-minioni10]
\definefontsynonym [MinionItalicCaps17]       [texnansi-axuc-minioni17]

\definefontsynonym [MinionSemiBoldCaps]       [texnansi-axuc-minionsb10]

\definefontsynonym [MinionSemiBoldItalicCaps] [texnansi-axuc-minionsbi10]
```

13. My thanks to Adam Lindsay for pointing this out.
14. Here is one example to get you started:

```
\definebodyfont
[24.9pt,20.7pt,17.3pt,14.4pt,12pt,11pt,10pt,9pt,8pt,7pt,6pt,5pt,4pt]
[rm]
[sp=SerifSemiBoldCaps sa 1,
 spa=SerifSemiBoldCaps scaled \magstep1, % or sa a
 spb=SerifSemiBoldCaps scaled \magstep2, % or sa b
 spc=SerifSemiBoldCaps scaled \magstep3, % or sa c
 spd=SerifSemiBoldCaps scaled \magstep4] % or sa d
```

and so forth.
15. That is, you will need to declare something like

```
\starttypescript[map] [minionu] [texnansi-axu]

\loadmapfile[texnansi-axu-adobe-minion.map]
\loadmapfile[texnansi-axuc-adobe-minion.map]

\stoptypescript
```

in `type-minu.tex`.
16. http://pragma-ade.com/general/magazines/mag-0009.pdf
17. I would like to especially thank Hans Hagen, Adam Lindsay, Thomas A.Schmitz, Ralf Stubner, and others from the ConTEXt mailing list for their help and assistance during the struggle to prepare this issue.
18. For an alternative approach, see Adam Lindsay's "OpenType installation basics for ConTEXt" in *The PracTEX Journal* 2005 No 02: http://tug.org/pracjourn/. It makes use of the `cfftot1` utility mentioned in footnote 9.

Idris Samawi Hamid