Bangla PDF HOWTO

Progga

<abulfazl AT juniv.edu>

2003−01−30

Revision History

Revision 1.1	2003−04−14	Revised by: Pro
PDF from Unicode, Caution about Bijoy		
Revision 1.0	2003−01−30	Revised by: Pro
Initial Release on Planet Earth, reviewed by LDP		

This text mainly describes the PDF creation process using KWord with the Bijoy2000 fonts and Bijoy keyboard, for working in the Bangla language. Some information on PDF creation from Unicode encoded Bangla text files is also provided.

Table of Contents

1. Introduction

PDF files, known for being viewable on most platforms, are a popular medium for distributing Bangla files over the Internet. Another use of PDF and Postscript files in the UNIX world is for printing. While PDF files can be created with various applications, this text is an attempt to describe the process of PDF creation using the free software KWord and ttf2pt1. The fonts used in this process are the Bijoy fonts included in the Bijoy2000 package. The keyboard layout for typing in Bangla is also Bijoy (the typing method of ligatures or compound characters is slightly different from the original one).

1.1. Copyright & License

1.2. Minimum Requirements

The test platform used were FreeBSD−4.6.0 RELEASE and RedHat−7.3. Earlier versions of RedHat won't do. Whatever is the platform, at least the following are required:

- XFree86−4.2.0 (unsure about earlier versions)
- KDE−3
- KWord−1.1.1
- TTF2PT1−3.4.3
- Bijoy2000 fonts
- Strong willingness to create a Bangla PDF.

There are many fonts that come with the Bijoy2000 package. In this text, to state a fontname, only "SutonnyMJ" has been used. This name can be replaced with others to meet individual needs.

1.3. Caution

After installing the Bijoy fonts and keyboard, one may be tempted to use them for purposes other than creating PDFs. But a little common sense and farsightedness will reveal that the use of a nonstandard font and keyboard like Bijoy is destined to create the same kind of chaos regarding Bangla in the Open Source environment as is now prevailing in the proprietary world. PDF is a different story though as they are font independent(if created properly) and has no chance of getting into this kind of chaos. Now what if one needs to create a Bangla text file? Easy − use Unicode. Recent developments has made it possible to write Unicode based Bangla in GTK+ 2.0 (or higher) based softwares. And the same thing is on the offing for QT (which is the backbone of KDE). Even PDFs can be created from Unicode encoded Bangla text files. This has been described in a later Section 6. So instead of relying on nonstandard softwares like Bijoy, try to use Unicode based Bangla wherever possible.

2. Fonts

While creating PDFs, KWord gives the best result if Type1 fonts are used. But Bangla Type1 fonts are quite rare, so converting TrueType fonts to Type1 is a good option. This conversion is done with the ttf2pt1 package. This package has some useful programs to carry out the conversion properly. After downloading and uncompression, the ttf2pt1 package can be installed using the following commands:

```
$ make all
$ make install
```

The last command needs to be executed as root. The `README` of ttf2pt1 has some better descriptions of these.

2.1. TrueType to Type1 Conversion

The following steps describe converting TrueType Bijoy fonts to Type1, suitable for PDF creation:

1. Instead of creating a Type1 font directly from `sutom___.ttf` (TrueType font file of SutonnyMJ font), create an interim file `sutonnymj.t1a`:
   ```
   $ ttf2pt1 sutom___.ttf sutonnymj
   ```
2. Perform some correction to `sutonnymj.t1a` using **forceiso**, a PERL script supplied with the ttf2pt1 package, and create `sutonnymj2.t1a`:
   ```
   $ cat sutonnymj.t1a | /usr/local/share/ttf2pt1/scripts/forceiso \
                      "U00%x" > sutonnymj2.t1a
   ```
   This correction is needed because some glyphs of Bijoy fonts have no name. **forceiso** solves this problem by assigning proper names to those otherwise nameless glyphs.
3. Finally, create the Type1 version of SutonnyMJ from `sutonnymj2.t1a` using **t1asm**, another program supplied with the ttf2pt1 package:
   ```
   $ cat sutonnymj2.t1a | /usr/local/libexec/ttf2pt1/t1asm \
                       > sutonnymj.pfa
   ```
   The last two steps can also be done in one go:
   ```
    $ cat sutonnymj.t1a | /usr/local/share/ttf2pt1/scripts/forceiso \
                "U00%x" | /usr/local/libexec/ttf2pt1/t1asm > sutonnymj.pfa
   ```

The final product, in this case `sutonnymj.pfa`, is the actual Type1 font file. The first step also produces `sutonnymj.afm` which can be ignored through the rest of the text.

2.2. Font Installation

There are some programs that automate font installation, such as type1inst for Type1, ttmkfdir for TrueType, KDE's own font installer for both types and some others. When these are used, the font encoding can be wrong and needs hand−editing of the `fonts.scale` file. If any of the font installing programs were used, **mkfontdir** must be executed after editing `font.scale` by hand. For Type1 fonts, the encoding is "adobe−fontspecific" and not "iso8859−1." For TrueType fonts, the encoding is "apple−roman" and not "iso8859−1". Besides, anti−aliasing must be stopped for these TrueType fonts to show up.

Alternatively, the next three sections can be considered to install fonts manually. This is a bit cumbersome but when things go wrong, it is easier to debug, as the user has a clear idea about the whole thing.

2.2.1. Type1 Font Installation

The following steps describe manual installation of Type1 fonts:

1. Create a new directory, `/usr/local/share/fonts/type1/bijoy/`, and copy the Type1 font there:

   ```
   $ mkdir −p /usr/local/share/fonts/type1/bijoy/
   $ cp sutonnymj.pfa /usr/local/share/fonts/type1/bijoy/
   ```
2. Create the file `fonts.scale` and place it inside `/usr/local/share/fonts/type1/bijoy/`. The `fonts.scale's` creation process will be described later in Section 2.2.2.
3. Create another file `fonts.dir` using **mkfontdir**:

   ```
   $ cd /usr/local/share/fonts/type1/bijoy/
   $ mkfontdir
   ```

   If the `fonts.scale` was created correctly then `fonts.dir` will be created, otherwise some error messages will appear after executing **mkfontdir**.
4. Open `XF86Config` and add `/usr/local/share/fonts/type1/bijoy/` as a new fontpath in the "Files" section:

   ```
   FontPath "/usr/local/share/fonts/type1/bijoy/"
   ```

   The `XF86Config` file is generally found inside `/etc/X11/`. If it's not present there then use **find** to get the path:

   ```
   $ find / −name XF86Config
   ```
5. Restart X server by pressing **Ctrl**+**Alt**+**Bksp**. The font will be installed.
6. Check whether the font has really been installed by using **xlsfonts**. This program shows the XLFD of every font that is available to the X server. **grep** can be used to find the desired font from **xlsfonts'** output:

   ```
   $ xlsfonts | grep sutonnymj
   ```

   If "sutonnymj" is unavailable to X then grep will give no output. Otherwise the output will be the XLFD of "sutonnymj":

   ```
         −altsys−sutonnymj−medium−r−normal−−0−0−0−0−p−0−adobe−fontspecific
   ```

   If more than one font were installed then **grep altsys** will show them all (if present in xlsfonts' output).

2.2.2. Creating `fonts.scale`

The `fonts.scale` file contains various information about the fonts in a directory where that `fonts.scale` itself is placed. This means every directory which has any font file can have its own `fonts.scale` (non−scalable fonts like Bitmap fonts do not need a `fonts.scale` file). The information that every line of a `fonts.scale` contains (preceded by the name of the font file itself) is called XLFD (X Logical Font Definition). An example line from a `fonts.scale` file is:

sutonny.pfa −altsys−SutonnyMJ−medium−r−normal−−0−0−0−0−p−0−adobe−fontspecific

The only difference is the very first line of `fonts.scale`, which has only a number instead of the usual font−file name followed by the corresponding XLFD. This number is the total number of XLFDs listed in that `fonts.scale` file.

The structure of a `fonts.scale` is like this:

```
 Line 1: Total number of XLFD
 Line 2: FontFileName1 XLFD
 Line 3: FontFileName2 XLFD
 ....
 ....
 ....
```

The following is a suitable XLFD for Type1 Bijoy fonts:

−altsys−SutonnyMJ−medium−r−normal−−0−0−0−0−p−0−adobe−fontspecific

Please note that in the above string, "SutonnyMJ" is the actual fontname. This name needs to be changed accordingly for other fonts.

An example `fonts.scale` file for two Type1 Bijoy fonts can be like this (Where sutonnymj.pfa and rinkymj.pfa are the actual Type1 font files for the fonts Sutonnymj and Rinkymj respectively):

```
2
sutonnymj.pfa −altsys−SutonnyMJ−medium−r−normal−−0−0−0−0−p−0−adobe−fontspecific
rinkymj.pfa −altsys−RinkyMJ−medium−r−normal−−0−0−0−0−p−0−adobe−fontspecific
```

It's better not to press **Return** after writing the XLFD in the last line. This sometimes create problem with the total number of XLFD lines and the file is taken as having bad structure by **mkfontdir**. If `fonts.scale` is not created properly, the **mkfontdir** command will give error message. Otherwise `fonts.dir` will be created.

The "adobe−fontspecific" substring found at the end of every XLFD for Type1 Bijoy fonts is the encoding of that font. If "iso8859−1" encoding is required, this can be done too by creating another file `fonts.alias`. Every line of `fonts.alias` contains two XLFDs. The first XLFD is the alias and the second one is the original. Unlike `fonts.scale` and `fonts.dir`, there is no number at the first line of `fonts.alias`. An example `fonts.alias's` *structure* looks like this:

```
Line 1: −altsys−SutonnyMJ−medium−r−normal−−0−0−0−0−p−0−iso8859−1
```

```
 −altsys−SutonnyMJ−medium−r−normal−−0−0−0−0−p−0−adobe−fontspecific
Line 2: −altsys−RinkyMJ−medium−r−normal−−0−0−0−0−p−0−iso8859−1
 −altsys−RinkyMJ−medium−r−normal−−0−0−0−0−p−0−adobe−fontspecific
```

Generally, some subdirectories under `/usr/X11R6/lib/X11/fonts/` contain many font files and some `fonts.scale` files, `fonts.dir` and `fonts.alias` as well. A browse through these files can make it easier to create new ones. One thing to notice is that the content of both `fonts.scale` and `fonts.dir` are same but both are still needed.

2.2.3. TrueType Font Installation

The Bijoy font's TrueType installation method is quite similar to that of Type1. Just keep a separate directory for the fonts, like,

`/usr/local/share/fonts/ttfonts/bijoy/`

and change the XLFD to:

−altsys−SutonnyMJ−medium−r−normal−−0−0−0−0−p−0−apple−roman

In the example, the font encoding is "apple−roman" instead of "adobe−fontspecific." Also to use these fonts anti−aliasing must be stopped, and this is where "Xft" comes into business.In your `$HOME` directory, create a hidden file `~/.xftconfig` and write the following lines in it (the file may be already present, in that case just add these lines):

```
dir "/usr/local/share/fonts/ttfonts/bijoy/"

match any family == "sutonnymj"
edit antialias = false; encoding = "apple−roman";
```

The `match any family == "sutonnymj"` and the next line prevents anti−aliasing for sutonnymj only. If there are other Bijoy fonts in use, more similar lines must be added.

The presence of only `~/.xftconfig` is enough to make a TrueType font available to KWord. There is no need to create `fonts.scale` and `fonts.dir`. Even the fontpath needs not be added in `XF86Config`. So these steps can be skipped if wished.

Newer systems(like RH 8.0) use Xft 2.0 instead of Xft 1.0 . Xft 2.0 doesn't use `~/xftconfig`. Instead it uses `~/.fonts` . This file can be modified by **fontconfig** . Here also, "antialias" must be stopped and "encoding" remains "apple−roman" .

2.3. On Using TrueType

KWord can't create PDFs perfectly using TrueType fonts, so there is no reason to use TrueType fonts for creating PDFs. But TrueType is useful for creating PDFs from files written in MS Word (i.e. *.doc files). Even then these PDFs are defective (due to the use of Type3 fonts inside the PDFs) and so are not transferable to other computers; they can be used for printing only. So the only suggested use of TrueType is to create

PDFs from MS Word files for printing. Again, these PDFs are of low quality, so if it is not urgently needed, TrueType should be avoided completely for PDF creation.

To open an MS Word file in KWord , the encoding of the relevant font(s) should be changed from "apple−roman" to "iso8859−1" in the `~/.xftconfig`, for example:

```
match any family == "sutonnymj"
    edit antialias = false; encoding = "apple−roman";
```

will become:

```
match any family == "sutonnymj"
    edit antialias = false; encoding = "iso8859−1";
```

Type1 and TrueType versions of a font should not be used together. To stop using TrueType fonts, the corresponding FontPath must be commented out from the "Files" section of `XF68Config`. The relevant entries for the font in `~/.xftconfig` must also be commented out. In both cases, a "#" as the first character on any line comments out the whole line. Given below is an example `~/.xftconfig`, where entry for a font has been commented out:

```
# dir "/usr/local/share/fonts/ttfonts/"

# match any family == "sutonnymj"
#           edit antialias = false; encoding = "apple−roman";
```

3. Keyboard

3.1. Using The Bijoy Keyboard

In X Window, all the keyboard related stuff is handled by XKB. XKB relies on some configuration files called the "symbol" files, to get the layout of a specific keyboard. The following steps describe the installation process of a symbol file for the Bijoy Bangla keyboard:

1. Below is the symbol file for the Bijoy keyboard. Save it as `bn_bijoy` .

```
// Symbol file for the Bijoy Bangla Keyboard.


partial default alphanumeric_keys
xkb_symbols "bijoy" {

name[group2]="Bangla";


key <AE01> { [], [     49,      exclam ] };
key <AE02> { [], [     50,      64  ]  };
key <AE03> { [], [     51,      35  ]  };
key <AE04> { [], [     52,      36  ]  };
key <AE05> { [], [     53,      37  ]  };
key <AE06> { [], [     54,      94  ]  };
key <AE07> { [], [     55,      117 ]  };
key <AE08> { [], [     56,      asterisk ] };
key <AE09> { [], [     57,      parenleft ] };
key <AE10> { [], [     48,      parenright ] };
key <AE11> { [], [     minus,   209        ] };
key <AE12> { [], [     61,      plus ]  };
key <AD01> { [], [     79,      115  ]  };
key <AD02> { [], [     104,     113  ]  };
key <AD03> { [], [     87,      88   ]  };
key <AD04> { [], [     99,      100  ]  };
key <AD05> { [], [     85,      86   ]  };
key <AD06> { [], [     80,      81   ]  };
key <AD07> { [], [     82,      83   ]  };
key <AD08> { [], [     110,     84   ]  };
key <AD09> { [], [     77,      78   ]  };
key <AD10> { [], [     111,     112  ]  };
key <AD11> { [], [     bracketleft,  braceleft ]};
key <AD12> { [], [     bracketright, braceright]};
key <AC01> { [], [     U84,     169  ]  };
key <AC02> { [], [     121,     126  ]  };
key <AC03> { [], [     119,     120  ]  };
key <AC04> { [], [     118,     65   ]  };
key <AC05> { [], [     Multi_key,      124  ]  };
key <AC06> { [], [     101,     102  ]  };
key <AC07> { [], [     75,      76   ]  };
key <AC08> { [], [     90,      95   ]  };
key <AC09> { [], [     96,      97   ]  };
key <AC10> { [], [  semicolon,  colon]  };
key <AC11> { [], [     213,     211  ]  };
key <TLDE> { [], [     212,     210  ]  };
key <BKSL> { [], [     114,     116  ]  };
key <AB01> { [], [     170,     168  ]  };
key <AB02> { [], [     73,      U8a  ]  };
key <AB03> { [], [     U87,     U89  ]  };
key <AB04> { [], [     105,     106  ]  };
key <AB05> { [], [     98,      89   ]  };
key <AB06> { [], [     109,     108  ]  };
key <AB07> { [], [     103,     107  ]  };
key <AB08> { [], [     comma,   less ]  };
key <AB09> { [], [     period,  greater]};
key <AB10> { [], [     slash,  question]};

} ;
```

2. Copy `bn_bijoy` to `/usr/X11R6/lib/X11/xkb/symbols/`
3. Edit the "InputDevice" section of `XF86Config` so it looks like:

```
.........
.........

Section "InputDevice"
        Identifier  "Keyboard0"
        Driver      "keyboard"

        Option   "XkbKeycodes"   "xfree86"
        Option   "XkbTypes"      "complete"
        Option   "XkbCompat"     "complete+leds"
        Option   "XkbSymbols"    "us(pc104)+bn_bijoy+group(lwin_toggle)"
        Option   "XkbGeometry"   "pc(pc104)"
EndSection
..........
..........
```

   The above description is for a 104−key keyboard.
4. Restart X server by pressing **Ctrl**+**Alt**+**Bksp**.
5. After restarting X, the Bijoy keyboard should be present along side the English one. To activate it, press the left **Win−key**. If everything is okay, the **Scroll Lock** LED will be on and the key presses will produce codes according to the Bijoy keyboard layout. Pressing the left **Win−key** again will disable Bijoy and the **Scroll Lock** LED will go off.
6. To test the newly installed Bijoy keyboard, follow these steps:
    1. Open KWord,
    2. Select sutonnymj from the font list,
    3. Press the left **Winkey**,
    4. Start typing according to the Bijoy keyboard layout.
7. A handy tool for testing mouse and keypress events is **xev**. This program shows all the generated codes from keypresses.

These steps are not enough, however, to write ligatures or compound characters. The next section describes this very thing.

3.2. Writing Ligatures

The ligature writing process described here is not a standard one . At best it can be called a work−around (it has a similarity to cuckoos laying eggs in crows' nests). If this method is used, the "Multikey" feature won't work on Latin characters (at least when Bijoy keyboard is needed). If there is no need to use the Multikey for typing various Latin characters like "ssharp" then this method is okay. The typing sequence of characters for writing ligatures is slightly different from the the original Bijoy keyboard. Whatever is the situation, the following steps describe a way to get the ligatures to appear on the screen:

1. Save the following as `Compose.bijoy`:

```
# Compose File for the Bijoy Bangla Keyboard
#
# Sequence Definition
#
# <Multi_key> Means <Compose>
# Special Character

# Special
<Multi_key> <Multi_key>                 : "\46"

# Ka
<Multi_key> <K> <K>                     : "\260"
<Multi_key> <O> <K>                     : "\274"
<Multi_key> <j> <K>                     : "\351"
<Multi_key> <l> <K>                     : "\256\213"
<Multi_key> <m> <K>                     : "\257\213"

# Kha

<Multi_key> <O> <L>                     : "\225\114"
<Multi_key> <m> <L>                     : "\366"

# Ga
<Multi_key> <M> <M>                     : "\271"
<Multi_key> <O> <M>                     : "\275"
<Multi_key> <grave> <M>                 : "\230\115"
<Multi_key> <j> <M>                     : "\352"
<Multi_key> <o> <M>                     : "\377"

#Gha
<Multi_key> <O> <N>                     : "\225\116"
<Multi_key> <grave> <N>                 : "\230\116"

# Cha
<Multi_key> <P> <P>                     : "\224\120"
<Multi_key> <T> <P>                     : "\302"
<Multi_key> <k> <P>                     : "\360"

# Chha
<Multi_key> <P> <Q>                     : "\224\121"
<Multi_key> <T> <Q>                     : "\303"

# Ja
<Multi_key> <R> <R>                     : "\276"
<Multi_key> <T> <R>                     : "\304"
<Multi_key> <e> <R>                     : "\342"

# Jha
<Multi_key> <R> <S>                     : "\300"
<Multi_key> <T> <S>                     : "\305"

# Io
<Multi_key> <R> <T>                     : "\301"

# Ta
<Multi_key> <K> <U>                     : "\261"
```

```
<Multi_key> <U> <U>                         : "\306"
<Multi_key> <Y> <U>                         : "\310"
<Multi_key> <b> <U>                         : "\233\125"
<Multi_key> <c> <U>                         : "\336"
<Multi_key> <j> <U>                         : "\353"
<Multi_key> <l> <U>                         : "\363"
<Multi_key> <m> <U>                         : "\367"

# Tha
<Multi_key> <Y> <V>                         : "\311"
<Multi_key> <b> <V>                         : "\332"
<Multi_key> <l> <V>                         : "\364"

# Da
<Multi_key> <W> <W>                         : "\307"
<Multi_key> <Y> <W>                         : "\312"
<Multi_key> <b> <W>                         : "\333"
<Multi_key> <j> <W>                         : "\354"

# Nna
<Multi_key> <l> <Y>                         : "\362"
<Multi_key> <n> <Y>                         : "\156\350"

# Ta
<Multi_key> <K> <Z>                         : "\263"
<Multi_key> <Z> <Z>                         : "\313"
<Multi_key> <b> <Z>                         : "\232\227"
<Multi_key> <c> <Z>                         : "\337"
<Multi_key> <m> <Z>                         : "\257\227"

# Tha
<Multi_key> <Z> <underscore>                : "\314"
<Multi_key> <b> <underscore>                : "\232\222"
<Multi_key> <m> <underscore>                : "\257\222"

# Dda
<Multi_key> <M> <grave>                     : "\272"
<Multi_key> <grave> <grave>                 : "\317"
<Multi_key> <b> <grave>                     : "\233\140"
<Multi_key> <e> <grave>                     : "\343"

# Dha
<Multi_key> <M> <a>                         : "\273"
<Multi_key> <grave> <a>                     : "\327"
<Multi_key> <b> <a>                         : "\334"
<Multi_key> <e> <a>                         : "\344"

# Na
<Multi_key> <K> <b>                         : "\113\350"
<Multi_key> <M> <b>                         : "\115\350"
<Multi_key> <N> <b>                         : "\116\350"
<Multi_key> <Y> <b>                         : "\131\350"
<Multi_key> <Z> <b>                         : "\132\350"
<Multi_key> <b> <b>                         : "\142\234"
<Multi_key> <c> <b>                         : "\143\350"
<Multi_key> <g> <b>                         : "\346"
<Multi_key> <k> <b>                         : "\153\350"
<Multi_key> <m> <b>                         : "\370"
<Multi_key> <n> <b>                         : "\375"

# Pa
<Multi_key> <c> <c>                         : "\340"
```

```
<Multi_key> <g> <c>                     : "\244\143"
<Multi_key> <j> <c>                     : "\355"
<Multi_key> <l> <c>                     : "\256\143"
<Multi_key> <m> <c>                     : "\257\143"

# Pha
<Multi_key> <g> <d>                     : "\347"
<Multi_key> <j> <d>                     : "\356"
<Multi_key> <l> <d>                     : "\365"
<Multi_key> <m> <d>                     : "\371"

# Ba
<Multi_key> <K> <e>                     : "\113\241"
<Multi_key> <M> <e>                     : "\115\246"
<Multi_key> <R> <e>                     : "\122\241"
<Multi_key> <U> <e>                     : "\125\241"
<Multi_key> <Y> <e>                     : "\131\136"
<Multi_key> <Z> <e>                     : "\132\241"
<Multi_key> <underscore> <e>            : "\137\241"
<Multi_key> <grave> <e>                 : "\330"
<Multi_key> <a> <e>                     : "\141\237"
<Multi_key> <b> <e>                     : "\233\136"
<Multi_key> <e> <e>                     : "\145\237"
<Multi_key> <g> <e>                     : "\244\136"
<Multi_key> <j> <e>                     : "\152\246"
<Multi_key> <k> <e>                     : "\153\246"
<Multi_key> <m> <e>                     : "\257\136"
<Multi_key> <n> <e>                     : "\156\237"

# Bha
<Multi_key> <grave> <f>                 : "\231\242"
<Multi_key> <g> <f>                     : "\244\242"

# Ma
<Multi_key> <K> <g>                     : "\264"
<Multi_key> <M> <g>                     : "\115\245"
<Multi_key> <O> <g>                     : "\225\147"
<Multi_key> <U> <g>                     : "\125\245"
<Multi_key> <Z> <g>                     : "\315"
<Multi_key> <grave> <g>         : "\331"
<Multi_key> <a> <g>                     : "\141\245"
<Multi_key> <b> <g>                     : "\142\245"
<Multi_key> <g> <g>                     : "\244\247"
<Multi_key> <j> <g>                     : "\152\245"
<Multi_key> <k> <g>                     : "\153\245"
<Multi_key> <l> <g>                     : "\256\247"
<Multi_key> <m> <g>                     : "\257\247"
<Multi_key> <n> <g>                     : "\376"

# La
<Multi_key> <K> <j>                     : "\113\254"
<Multi_key> <M> <j>                     : "\115\254"
<Multi_key> <c> <j>                     : "\143\254"
<Multi_key> <d> <j>                     : "\144\254"
<Multi_key> <e> <j>                     : "\145\254"
<Multi_key> <f> <j>                     : "\146\254"
<Multi_key> <g> <j>                     : "\244\254"
<Multi_key> <j> <j>                     : "\152\254"
<Multi_key> <l> <j>                     : "\153\254"
<Multi_key> <m> <j>                     : "\257\254"
<Multi_key> <n> <j>                     : "\156\254"
```

```
# Ssa
<Multi_key> <K> <l>                         : "\266"

# Sa
<Multi_key> <K> <m>                         : "\267"
<Multi_key> <b> <m>                         : "\335"
<Multi_key> <c> <m>                         : "\341"

# Miscellaneous
<Multi_key> <K> <ordfeminine>               : "\265"
<Multi_key> <Z> <ordfeminine>               : "\316"
<Multi_key> <f> <ordfeminine>               : "\345"
<Multi_key> <n> <U84>                       : "\374"
<Multi_key> <M> <y>                         : "\270"
<Multi_key> <i> <y>                         : "\151\223"
<Multi_key> <k> <y>                         : "\357"
<Multi_key> <n> <y>                         : "\373"
<Multi_key> <i> <asciitilde>                : "\151\203"

# Vowels
<Multi_key> <w>                             : "\102"
<Multi_key> <x>                             : "\103"
<Multi_key> <y>                             : "\104"
<Multi_key> <y>                             : "\104"
<Multi_key> <asciitilde>                    : "\105"
<Multi_key> <U84>                           : "\106"
<Multi_key> <U87>                           : "\107"
<Multi_key> <U89>                           : "\110"
<Multi_key> <U8a>                           : "\112"


# End of Sequence Definition
```

2. Get the current locale name. One way to get it is:

```
$ echo $LANG
```

3. Find the `Compose` file for the current locale. `/usr/X11R6/lib/X11/locale/compose.dir` lists the `Compose` files for all the locales. If the locale is "C", the `Compose` file is `iso8859−1/Compose`, i.e. `/usr/X11R6/lib/X11/locale/iso8859−1/Compose`. If the locale is "en_US.ISO8859−15," the `Compose` file is `iso8859−15/Compose`, i.e. `/usr/X11R6/lib/X11/locale/iso8859−1/Compose`. Whichever is the Compose file for the current locale, make a backup of it:

```
$ cd /usr/X11R6/lib/X11/locale/iso8859−1/
$ mv Compose Compose.real
```

   The above example commands were written assuming that the locale was "C."
4. Make `Compose.bijoy` the new `Compose` file for the current locale:

```
$ cp Compose.bijoy /usr/X11R6/lib/X11/locale/iso8859−1/Compose
```

   The above example command was written assuming that the locale was "C."
5. The ligatures can be written now. To test it do the following:
    1. Open KWord,
    2. Select sutonnymj from the font list,
    3. Press the left **Winkey**,
    4. Press 'Hashanta'(g)+'Ka'(j)+'Ka'(j).

If the output is Zukta Ka, then the ligature writing process is okay. Now the thing to notice here is that, unlike the original Bijoy keyboard, Hashanta has been pressed before the two 'Ka's. The original Bijoy keyboard requires that 'Hashanta' be pressed in the midst of the two 'Ka's, for example:

Zukta Ka = 'Ka'(j)+'Hashanta'(g)+'Ka'(j)

Except this change of sequence for writing ligatures there are a few minor ones that have been listed below:

Original Sequence	**Changed Sequence**
'Ka'(j)+'Rafala'(z)	'Hashanta'(g)+'Ka'(j)+'Rafala'(z)
'Ta'(k)+'Rafala'(z)	'Hashanta'(g)+'Ta'(k)+'Rafala'(z)
'Va'(H)+'Rafala'(z)	'Hashanta'(g)+'Va'(H)+'Rafala'(z)
'Ha'(i)+'Rikar'(a)	'Hashanta'(g)+'Ha'(i)+'Rikar'(a)
'Ga'(o)+'Ukar'(s)	'Hashanta'(g)+'Ga'(o)+'Ukar'(s)
'Ra'(v)+'Ukar'(s)	'Hashanta'(g)+'Ra'(v)+'Ukar'(s)
'Sha'(M)+'Ukar'(s)	'Hashanta'(g)+'Sha'(M)+'Ukar'(s)
'Ha'(i)+'Ukar'(s)	'Hashanta'(g)+'Ha'(i)+'Ukar'(s)
'Ra'(v)+'UUkar'(S)	'Hashanta'(g)+'Ra'(v)+'UUkar'(S)

4. Printing as PDF

After typing some text in KWord, do the following to get a PDF:

1. Press Print Preview; if the output is okay, the PDF will be okay. If not, the font conversion was faulty, so check it again (especially the use of forceiso).
2. Word−wrapping in the "Preview" window and in KWord may look different. If this is unaccptable then it needs to be adjusted manually. The PDF will always appear as shown in the "Preview."
3. Press Print and a box will appear. Press System Options... and select Embed fonts in PostScript data when printing. Then click OK.
4. From the printer selection list select Print To File (PDF/Acrobat). Change the output file's name or leave it unchanged.
5. Now press Print. It may take a while for the box to disappear and the PDF to be created.
6. Now check the PDF with any PDF viewer.
7. A good utility to use here is **pdffonts** to check whether the Bangla font has really been embeded in the newly created PDF.

5. RedHat Peculiarities

In some places, the previous descriptions are not applicable to RedHat. These differences are described below:

1. While installing new font(s), the fontpath was supposed to be included in the "Files" section of `XF86Config`. In the case of RedHat this is not needed at all. There is a script called **chkfontpath** to do this. With **chkfontpath**, add a fontpath like this:

   ```
   $ chkfontpath −a /usr/local/share/fonts/ttfonts/bijoy/
   ```

   And remove a fontpath like this:

   ```
   $ chkfontpath −r /usr/local/share/fonts/ttfonts/bijoy/
   ```

   If **chkfontpath** doesn't show any error messages then the font(s) become available or unavailable just after executing it. But even for RedHat, there is no need to add a TrueType fontpath as "Xft" itself makes the fonts avilable to KWord. So use of **chkfontpath** for TrueType is optional.

   Note

   Just for reference, in the case of RedHat, the fontpaths are added in `/etc/X11/fs/config` and not in `XF86Config` or `XF86Config−4`.
2. To use the Bijoy keyboard, the "InputDevice" section of `XF86Config` needs editing. For RedHat it's a different file, `XF86Config−4` (i.e. `/etc/X11/XF86Config−4`).
3. The only use of TrueType was stated as low quality PDF creation from MS Word files, for printing only. Unfortunately KWord failed to open any MS Word files in RedHat, let alone create PDFs. Don't know why.

6. PDF from Unicode

6.1. Lekho

Lekho is a simple text editor with the capability to create Unicode encoded Bangla+English text files. It uses the "Adarshalipi" family of TrueType fonts for showing Bangla glyphs. If you have never used it then have a look at it's website. The website along with the Lekho distribution contain enough docs to get someone start using Lekho.

The "Adarshalipi" fonts are quite similar to the Bijoy fonts and so the Type1 conversion procedure is same as the Bijoy fonts. Using a Type1 Adarshalipi font, Lekho can produce PDFs from Unicode encoded Bangla text files. Actually Lekho produces Postscript files which in turn is converted to PDF using tools like **ps2pdf**. The next steps describe PDF creation using Lekho, provided that a Type1 "Adarshalipi" font has been installed already:

1. When editing is over, change the font size of both Bangla and English fonts to 11. This is not mandatory but it keeps the format of the Postscript file as near as it is seen in Lekho.
2. Click the PRINT FILE button and select PRINT TO FILE. Write a name for the soon to be created Postscript file and click OK.
3. Use a tool like **ps2pdf** to convert the Postscript file to PDF:

```
        $ ps2pdf file.ps
```

Lekho has another valuable feature − it can export a file to bangtex , the Latex macro package for Bangla. So, the Latex users can eventually use this feature to create PDFs.

6.2. BSpeller

BSpeller is basically a Bangla spell checker. Besides, it is a light weight text editor with the ability to print. Instead of TrueType or Type1, BSpeller relies on OpenType fonts. So it requires GTK+ 2.0 (or later) to render Bangla glyphs. As it is still a beta software, it's output is somewhat shaky.

7. Other Resources

7.1. Useful tools

The following is a list of programs that may be useful for debugging any trouble regarding the fonts, keyboard or the PDFs. Their usages are well documented in their respective man pages:

- **xlsfonts**, lists the XLFDs of the all fonts available to X.
- **xfd**, shows all the glyphs of a font when available to X.
- **xset**, resets various X options like fontpath.
- **xev**, detects and shows various mouse and keypress events.
- **xkbcomp**, besides other operations, prints the current XKB configuration in a single file.
- **xpdf**, a PDF viewer.
- **pdffonts**, shows font list of a PDF file.
- **ps2pdf**, PostScript to PDF converter.

7.2. Useful Links & References

- A detailed description of XKB, www.charvolant.org/~doug/xkb/
- A one stop mall for Unicode based Bangla on Linux, www.bengalinux.org
- The XFree86−De−uglification−Mini−HOWTO
- Font−HOWTO
- Fonts in XFree86, also supplied with the XFree86 documentation as `README.fonts` and normally found inside `/usr/X11R6/lib/X11/doc/`
- Bangla Localisation HOWTO, a good starting point for using Unicode based Bangla.

8. Acknowledgments

- Jan Benedict Glaw (jbglaw AT lug−owl.de), for advocating Bangla in spite of being non−Bangali.
- Shishir Bhai (shishir_faruk AT yahoo.com) + Dolon (mrk_ju AT yahoo.com) &Co, for the inspiration.
- Mojahed Bhai (mojahed AT agni.com), for being the Guinea pig ;−)
- Sergey Babkin (babkin AT bellatlantic.net), the ultimate Type1 guy.
- People at the Ankur (formerly Bengalinux) group, for working towards the future.
- Tabatha Persad (tabatha AT merlinmonroe.com), for the review and valuable changes.
- To the Almighty, for this text wouldn't have been possible without his (???) will.