pst-barcode

A PSTricks package for drawing bar codes; v.0.14

June 17, 2015

Documentation by
Herbert Voß

Package author(s):
Terry Burton
Herbert Voß
Contents

1 Introduction 4

2 The options 4
  2.1 The \TeX{} options 4
  2.2 The PostScript options 5
  2.3 Examples for the \TeX{} options 5
  2.4 Examples for the PostScript options 7

3 Usage 7

4 Possible barcodes 8
  4.1 EAN-13 8
  4.2 EAN-8 9
  4.3 UPC-A 9
  4.4 UPC-E 9
  4.5 EAN-5 10
  4.6 EAN-2 10
  4.7 ISBN 10
  4.8 Code-39 11
  4.9 Code-128 and UCC/EAN-128 11
  4.10 Rationalized Codabar 12
  4.11 Interleaved 2 of 5 and ITF-14 13
  4.12 Code 2 of 5 13
  4.13 Postnet 14
  4.14 Royal Mail 14
  4.15 Kix (Customer index) – Dutch Mail 14
  4.16 Australian postal service 15
  4.17 Japan post service 15
  4.18 onecode 15
  4.19 Symbol 16
  4.20 MSI 16
  4.21 Plessey 16
  4.22 Reduced Space Symbology (RSS) 17
  4.23 Pharmacode 17
  4.24 PDF417 18
  4.25 Data matrix 18
  4.26 2D Maxi code 19
  4.27 Aztec Code 21
  4.28 itf14 21
  4.29 QR Code 21

5 Code Commentary 24
  5.1 The Barcode Data Structure 24
  5.2 An Encoder 25
  5.3 The Renderer 27
5.4 Notes Regarding Coding Style ........................................... 29
5.5 Installing the Barcode Generation Capability into a Printer’s Virtual Ma-
    chine ................................................................. 29

6 QR code in the margin ..................................................... 30

7 List of all optional arguments for pst-barcode ......................... 31

References ........................................................................ 31
The `pstricks` related package provides (essentially) one macro for printing barcodes. The type of the code is defined by a parameter and passed to postscript.

## 1 Introduction

The `pstricks` package provides (essentially) one macro for printing barcodes. The type of the code is defined by a parameter and passed to postscript. To install the package put the three files in a place, where \TeX will search for the files:

<table>
<thead>
<tr>
<th>name</th>
<th>meaning</th>
<th>target dir</th>
</tr>
</thead>
<tbody>
<tr>
<td>pst-barcode.tex</td>
<td>\LaTeX style file – wrapper</td>
<td>$\text{LOCALTEXMF}/\text{tex/generic/pstricks/}$</td>
</tr>
<tr>
<td>pst-barcode.sty</td>
<td>\TeX file – PS interface</td>
<td>$\text{LOCALTEXMF}/\text{latex/pstricks/}$</td>
</tr>
<tr>
<td>pst-barcode.pro</td>
<td>PostScript file</td>
<td>$\text{LOCALTEXMF}/\text{dvips/pstricks/}$</td>
</tr>
<tr>
<td>pst-barcode-doc.tex</td>
<td>documentation source</td>
<td>$\text{LOCALTEXMF}/\text{doc/pstricks/}$</td>
</tr>
<tr>
<td>pst-barcode-doc.bib</td>
<td>bibliography source</td>
<td>$\text{LOCALTEXMF}/\text{doc/pstricks/}$</td>
</tr>
<tr>
<td>pst-barcode-doc.pdf</td>
<td>documentation</td>
<td>$\text{LOCALTEXMF}/\text{doc/pstricks/}$</td>
</tr>
</tbody>
</table>

There is only one macro `\psbarcode` with the usual PStricks syntax

\begin{verbatim}
\psbarcode [Options] {text or filename}{PS options}{barcode type}
\end{verbatim}

Important is the fact, that the barcode is printed in a \TeX box of zero dimension. If you want to save some space in your text, use the `pspicture` environment or the `\makebox` macro.

## 2 The options

### 2.1 The \TeX options

<table>
<thead>
<tr>
<th>name</th>
<th>default</th>
<th>remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>transx</td>
<td>0</td>
<td>horizontal shift</td>
</tr>
<tr>
<td>transy</td>
<td>0</td>
<td>vertical shift</td>
</tr>
<tr>
<td>scalex</td>
<td>1</td>
<td>horizontal scaling</td>
</tr>
<tr>
<td>scaley</td>
<td>1</td>
<td>vertical scaling</td>
</tr>
<tr>
<td>rotate</td>
<td>0</td>
<td>rotating angle in degrees</td>
</tr>
<tr>
<td>file</td>
<td>false</td>
<td>load an external file for the text</td>
</tr>
</tbody>
</table>
2.2 The PostScript options

<table>
<thead>
<tr>
<th>name</th>
<th>default</th>
<th>remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>1</td>
<td>dimension is inch</td>
</tr>
<tr>
<td>textsize</td>
<td>10</td>
<td>dimension is pt</td>
</tr>
<tr>
<td>textpos</td>
<td>-2</td>
<td>dimension is pt; it is the shift for additional code text</td>
</tr>
<tr>
<td>inkspread</td>
<td>0.15</td>
<td>dimension is pt</td>
</tr>
<tr>
<td>showborder</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>borderwidth</td>
<td>0.5</td>
<td>dimension in pt</td>
</tr>
<tr>
<td>borderleft</td>
<td>10</td>
<td>dimension in pt</td>
</tr>
<tr>
<td>borderright</td>
<td>10</td>
<td>dimension in pt</td>
</tr>
<tr>
<td>borderleft</td>
<td>10</td>
<td>dimension in pt</td>
</tr>
<tr>
<td>borderbottom</td>
<td>1</td>
<td>dimension in pt</td>
</tr>
<tr>
<td>borderwidth</td>
<td>0.5</td>
<td>dimension in pt</td>
</tr>
<tr>
<td>width</td>
<td>-</td>
<td>dimension in inch</td>
</tr>
<tr>
<td>font</td>
<td>/Helvetica</td>
<td>must be a PostScript font</td>
</tr>
<tr>
<td>includetext</td>
<td>-</td>
<td>enable human readable text</td>
</tr>
<tr>
<td>includecheck</td>
<td>-</td>
<td>enable check digit</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>-</td>
<td>check digit visible in text</td>
</tr>
<tr>
<td>parse</td>
<td>-</td>
<td>parse variable field für decimal values, like `032 for space, and convert them to ASCII</td>
</tr>
</tbody>
</table>

2.3 Examples for the TeX options

\psframebox{\begin{pspicture}(2.5,1 in)\psbarcode{01335583}{includetext}{ean8}\end{pspicture}}quad\psframebox{\begin{pspicture}(-2.6,-1.5)(0.4,0.2 in)\psbarcode[rotate=180, linecolor=red]{01335583}{includetext guardwhitespace height=0.6}{ean8}\end{pspicture}}quad\psframebox{\begin{pspicture}(3.8,1 in)\psbarcode[ scalex=1.5, scaley=0.5, transy=1]{01335583}{includetext inkspread=0.5}{ean8}\end{pspicture}}
The contents of the external file demo.tex:

```
\documentclass{article}
\usepackage{pst-barcode,fancyvrb}
\begin{document}
\VerbatimInput{\jobname.tex}% test
\begin{pspicture}(2in,2in)
  \psbarcode[file]{\jobname.tex}{qrcode}
\end{pspicture}
\end{document}
```
2.4 Examples for the PostScript options

\begin{pspicture}(3.5,1.2in)
\psbarcode{01335583}{includetext guardwhitespace height=0.6}{ean8}
\end{pspicture}

\begin{pspicture}(3.5,1.2in)
\psbarcode{01335583}{textsize=15 includetext guardwhitespace height=0.6}{ean8}
\end{pspicture}

\begin{pspicture}(3.5,1.2in)
\psbarcode{01335583}{includetext inkspread=0.5}{ean8}
\end{pspicture}

\begin{pspicture}(3.5,1.2in)
\psbarcode{01335583}{includetext textpos=0}{ean8}
\end{pspicture}

\begin{pspicture}(3.5,1.2in)
\psbarcode{01335583}{includetext guardwhitespace}{ean8}
\end{pspicture}

\begin{pspicture}(3.5,1.2in)
\psbarcode{01335583}{textsize=15 includetext guardwhitespace width=2}{ean8}
\end{pspicture}

3 Usage

By default the barcode has a width and a height of zero. Using the \parbox macro or the \pspicture environment can reserve the needed space for the barcode. The \fbox in the following examples is used only for demonstration.
4 Possible barcodes

The following section shows the symbologies that are supported by the encoders, including the available features for each. This list may not be up-to-date. If it does not contain any of the formats or features that you require then check the project source code or try the support mailing list.

4.1 EAN-13

Characters 0123456789

Data 12 or 13 digits

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>

Notes If just 12 digits are entered then the check digit is calculated automatically.
4.2 EAN-8

Characters 0123456789

Data 8 digits

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>

\begin{pspicture}(-2,-1.2)(0,0.2in)
\psbarcode[rotate=180,linecolor=red]{01335583}{includetext guardwhitespace height=0.6}{ean8}
\end{pspicture}

4.3 UPC-A

Characters 0123456789

Data 11 or 12 digits

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>

Notes If just 11 digits are entered then the check digit is calculated automatically

\begin{pspicture}(3,1.2in)
\psbarcode[transx=15pt,transy=10pt]{78858101497}{includetext}{upca}
\qdisk(0,0){3pt}\rput[lb](5pt,-10pt){Origin}
\end{pspicture}

4.4 UPC-E

Characters 0123456789

Data 7 or 8 digits

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>

Notes If just 7 digits are entered then the check digit is calculated automatically

\begin{pspicture}(3,1.2in)
\psbarcode[transx=15pt,transy=10pt]{78858101497}{includetext}{upca}
\end{pspicture}
4.5 EAN-5

Characters 0123456789

Data 5 digits

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>

4.6 EAN-2

Characters 0123456789

Data 2 digits

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>

4.7 ISBN


Characters -0123456789

Data 9 or 10 digits for ISBN-10 seperated appropriately with dashes
4.8 Code-39

Data 12 or 13 digits for ISBN-13 separated appropriately with dashes

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>

Notes If just 9 (ISBN-10) or 12 (ISBN-13) digits are entered then the human readable, ISBN check digit is calculated automatically

4.9 Code-128 and UCC/EAN-128

Characters !"#$%&'()*+,-./0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ-. $/

Data Variable number of ASCII characters and special function symbols, starting with the appropriate start character for the initial character set. UCC/EAN-128s must have a mandatory FNC 1 symbol immediately following the start character.
### Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>parse</td>
<td>Any instances of ^NNN in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters, e.g. ^029 for GS, etc.</td>
</tr>
<tr>
<td>parsefnc</td>
<td>Non-data function characters like ^FNC1. The special pseudo characters ^LNKA and ^LNKC at the end of the symbol indicate that a GS1-128 symbol includes a CC-A/B or CC-C GS1 composite 2D component.</td>
</tr>
<tr>
<td>raw</td>
<td>The data field provides the input as pre-encoded codewords in ^NNN format, suitable for direct low-level encoding.</td>
</tr>
</tbody>
</table>

### Notes

Any non-printable character can be entered via its escaped ordinal value, for example ^070 for ACK and ^102 for FNC 1. Since a caret symbol serves as an escape character it must be escaped as ^062 if used in the data. The check character is always added automatically.

```
\begin{pspicture}(5,1in)
\psbarcode{"104^102Count^0991234^101!}{includetext}{code128}
\end{pspicture}
```

```
\begin{pspicture}(5,1in)
\psbarcode{"^FNC3L09^}{parsefnc includetext}{code128}
\end{pspicture}
```

### 4.10 Rationalized Codabar

**Characters** 0123456789-$/:./+ABCD

**Data** Variable number of digits and any of the symbols -$/:./+ABCD.

### Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includecheck</td>
<td>Enable check digit</td>
</tr>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>Make check digit visible in text</td>
</tr>
</tbody>
</table>
4.11 Interleaved 2 of 5 and ITF-14

**Characters** 0123456789

**Data** Variable number of digits. An ITF-14 is 14 characters and does not have a check digit.

**Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includecheck</td>
<td>Enable check digit</td>
</tr>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>Make check digit visible in text</td>
</tr>
</tbody>
</table>

**Notes** The data may be automatically prefixed with 0 to make the data, including optional check digit, of even length.

4.12 Code 2 of 5

**Characters** 0123456789

**Data** Variable number of digits

**Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>
4.13 Postnet

Characters 0123456789

Data Variable number digits

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>Make the check digit visible in the text</td>
</tr>
</tbody>
</table>

Notes Check digit is always added automatically

\begin{pspicture}(7,0.3in)
\psbarcode{01234}{includetext textpos=-10 textfont=Helvetica textsize=10}{postnet}
\end{pspicture}

4.14 Royal Mail

Characters ZUVWXY501234B6789AHCEFNIJKLMTOPQRS

Data Variable number digits and capital letters

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>Make the check digit visible in the text</td>
</tr>
</tbody>
</table>

Notes Check digit is always added automatically

\begin{pspicture}(5,0.5in)
\psbarcode{LE28HS9Z}{includetext}{royalmail}
\end{pspicture}

4.15 Kix (Customer index) – Dutch Mail

Characters ZUVWXY501234B6789AHCEFNIJKLMTOPQRS

Data Variable number digits and capital letters

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
</tbody>
</table>

Notes Check digit is always added automatically

\begin{pspicture}(5,0.3in)
\psbarcode{1203AA12}{includetext}{kix}
\end{pspicture}
4.16 Australian postal service

Characters ZUWXY501234B6789AHCDEFGNIJKLMTOPQRSabc...xyz

Data Variable number digits and letters

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>Make the check digit visible in the text</td>
</tr>
</tbody>
</table>

\begin{pspicture}(5,0.3in)
\psbarcode{1139549554}\{includetext\}\{auspost\}
\end{pspicture}

4.17 Japan post service

Characters ZUWXY501234B6789AHCDEFGNIJKLMTOPQRSabc...xyz

Data Variable number digits and letters

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>Make the check digit visible in the text</td>
</tr>
</tbody>
</table>

\begin{pspicture}(0,-5mm)(7,0.5in)
\psbarcode{6540123789-A-K-Z}\{includetext textxalign=center\}\{japanpost\}
\end{pspicture}

4.18 onecode

Characters 0123456789

Data Variable number digits

\begin{pspicture}(0,-5mm)(5,0.3in)
\psbarcode{0123456709498765432101234567891}\{includetext\}\{onecode\}
\end{pspicture}
4.19 Symbol

The purpose of the symbol encoder is to store the definitions of miscellaneous barcode symbols such as the FIM symbols used by the US Postal Service on their reply mail.

\begin{pspicture}(1cm,1.5cm)
\psbarcode{fima}()
\end{pspicture}

\begin{pspicture}(1cm,1.5cm)
\psbarcode{fimb}()
\end{pspicture}

\begin{pspicture}(1cm,1.5cm)
\psbarcode{fimc}()
\end{pspicture}

\begin{pspicture}(1cm,1.5cm)
\psbarcode{fimd}()
\end{pspicture}

4.20 MSI

Characters 0123456789

Data Variable number digits

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includecheck</td>
<td>Enable check digit</td>
</tr>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>Make check digit visible in the text</td>
</tr>
</tbody>
</table>

\begin{pspicture}(6,1in)
\psbarcode{0123456789}{includecheck includetext}{msi}
\end{pspicture}

4.21 Plessey

Characters 01234B6789abcdef

Data Variable number of hexadecimal characters

Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>includetext</td>
<td>Enable human readable text</td>
</tr>
<tr>
<td>includecheckintext</td>
<td>Make the check digits visible in the text</td>
</tr>
</tbody>
</table>

\begin{pspicture}(6,1in)
\psbarcode{01234B6789abcdef}{includecheck includetext}{plessey}
\end{pspicture}
Notes  Check digits are always added automatically.

4.22 Reduced Space Symbology (RSS)

This is a family that includes RSS-14®, RSS Limited®, and RSS Expanded® (http://www.gs1.org/productssolutions/barcodes/technical/rss.html). RSS-14 and RSS Limited encode Global Trade Item Numbers (GTINs). RSS Expanded will encode any of the GS1 System identification numbers as well as all other Application Identifiers. RSS is used in the healthcare industry.

**Characters**  0123456789

**Data**  Variable number digits

**rss14 (databaromni)**

\begin{pspicture}(12,.3in)
\psbarcode{(01)24012345678905}{format=truncated includetext height=0.3}{databaromni}
\end{pspicture}

**rsslimited (databarlimited)**

\begin{pspicture}(12,.3in)
\psbarcode{(01)15012345678907}{height=0.3}{databarlimited}
\end{pspicture}

**rssexpanded (databarexpanded)**

\begin{pspicture}(12,.3in)
\psbarcode{(10)12A}{height=0.3}{databarexpanded}
\end{pspicture}

4.23 Pharmacode


**Characters**  0123456789
Data Variable number digits

```
\begin{pspicture}(12,.3in)
\psbarcode{117480}\{(pharmacode)\}
\end{pspicture}
```

4.24 PDF417


Characters 0123456789

Data Variable number characters

```
\begin{pspicture}(2in,0.3in)
\psbarcode{^453^178^121^239}\{(columns=2 rows=10)\}{pdf417}
\end{pspicture}
```

4.25 Data matrix

For a documentation see [http://de.wikipedia.org/wiki/Strichcode#DataMatrix](http://de.wikipedia.org/wiki/Strichcode#DataMatrix). It is a 2D matrix-style barcode that can encode full 256 character extended-ASCII. Also known as: Data Matrix ECC 200. Variants:

**GS1** DataMatrix is a variant of Data Matrix that should be used when encoding data that is in GS1 Application Identifier standard format.

**HIBC** Data Matrix is a variant of Data Matrix that should be used when encoding HIBC formatted data.

Standards: ISO/IEC 16022, ANSI/AIM BC11 ISS.

Data and Options

- The data field can contain any extended ASCII data.
- When the parse option is specified, any instances of \(^NNN\) in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.
- When the parsefnc option is specified, non-data function characters can be specified by \(^FNC1\).
- The columns and rows options are used to specify the size of the symbol, either square or rectangular; one of: Square: 10x10, 12x12, 14x14, 16x16, 18x18, 20x20, 22x22, 24x24, 26x26, 32x32, 36x36, 40x40, 44x44, 48x48, 52x52, 64x64, 72x72, 80x80, 88x88, 96x96, 104x104, 120x120, 132x132, 144x144 Rectangular: 8x18, 8x32, 12x26, 12x36, 16x36, 16x48
- If the columns and rows are unspecified, the encoder will default to creating a (non-rectangular) symbol that is the minimum size to represent the given data.
- The raw option denotes that the data field is providing the input as a pre-encoded codewords in ^NNN format, suitable for direct low-level encoding.
4.26 2D Maxi code

For a documentation see http://www.logicalconcepts.eu/wDeutsch/autoid/barcodetypen/index.html?navid=21. MaxiCode is a fixed-sized two-dimensional symbology created by the United Parcel Service that is primarily used for freight sortation and tracking. Its symbols have modules arranged in a hexagonal grid around a circular finder pattern which can be read omnidirectionally.

- MaxiCode has five alphabets A, B, C, D and E, each containing 64 characters.
- Alphabet A contains mostly upper case letters, numbers and some common ASCII symbols.

- Alphabet B contains mostly lower case letters and common ASCII symbols.

- Alphabet C contains mostly upper case letters from the extended ASCII character set and less common ASCII symbols.

- Alphabet D contains mostly lower case letters from the extended ASCII character set and less common ASCII symbols.

- Alphabet E contains mostly the special ASCII characters and unprintable symbols.

- Non-printable/typable characters can be entered as their escaped ordinal values, e.g. ^028 for FS and ^059 for [shift B], etc.

- The symbol always starts in alphabet A which is suitable for the most basic contents.

- You can switch the working alphabet within the data using the [latch B], [latch A], ... characters.

- You can temporarily shift to another alphabet for a varying number of characters using the [shift A], [2 shift A], [3 shift A], ... characters.

- You can remain in an alphabet to which you have shifted using the [lock in C], [lock in D], ... characters.

- There are also more advanced features for encoding for which a thorough reading of the specification is required as these cannot be described succinctly here.

The modes:

- Mode 0 - Obsolete. (Older printers will produce Mode 0 if the firmware is outdated. Mode 0 MaxiCodes can be visually determined by examining the two horizontal hexagons in the upper right-hand corner. They will be white if the Mode is 0. For all other modes, they are black.)

- Mode 2 - Used for Numeric postal codes. (Primary use is US domestic destinations.)

- Mode 3 - Used for Alphanumeric postal codes. (Primary use is Int’l destinations.)

- Mode 4 - Standard Error Correction.

- Mode 5 - Enhanced Error Correction.

- Mode 6 - Used for programming hardware devices.

Characters @ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

Data Variable number characters
4.27 Aztec Code

Aztec Code is a 2 dimensional matrix style bar code symbology. Aztec Code was invented by Andrew Longacre (USA) in 1995 (http://de.wikipedia.org/wiki/Aztec_Code).

Characters 0123456789

Data Variable number characters

\begin{pspicture}(1in,1in)
\expandafter\psbarcode{\string}>^29906^1523828^029840^029001^029159^29\text{Z}^0004951^029^UPSN^02906^X^29\text{610}^0291234567^0291/1^029^029Y^029634\text{ALPHA DR}^029PITTSBURGH^029PA^029^004}{\text{mode}=2\text{ parse}}{\text{maxicode}}\end{pspicture}

\begin{pspicture}(1in,1in)
\psbarcode{0123456789}\% {\text{format}=\text{compact}} {\text{layers}=3}\text{azteccode}\end{pspicture}

4.28 itf14

Characters 0123456789

Data Variable number characters

\begin{pspicture}(1in,1in)
\psbarcode{1001234567890}{\text{showborder borderwidth}=4\text{ borderleft}=15\text{ borderright}=15\text{ height}=0.5\text{ includecheck includetext includecheckintext textyoffset=-10}\text{interleaved2of5}\end{pspicture}

4.29 QR Code


Characters 0123456789
D ata  Variable number characters

The data field can contain any extended ASCII data. When the parse option is specified, any instances of \^{N\text{NN}} in the data field are replaced with their equivalent ASCII value, useful for specifying unprintable characters.

The eclevel option is used to specify the error correction level:

- eclevel=L - Low (default for micro format symbols)
- eclevel=M - Medium (default for full format symbols)
- eclevel=Q - Quality
- eclevel=H - High

The eclevel will be opportunistically raised when this does not result in an increased symbol size. The encoding option is used to specify a desired encoding for the input data which can sometimes result in a more optimal symbol size:

- encoding=alphanumeric - Alphanumeric data
- encoding=numeric - Numeric data
- encoding=byte - Byte based encoding
- encoding=kanji - Kanji characters based on Shift JIS

**encoding=raw** - Equivalent to the raw option If left unspecified the optimal available encoding will be chosen for the given data in the following order of preference: numeric, alphanumeric, kanji, byte. The version option is used to specify the size of the symbol, 1 to 40 for full format symbols or version=M1, version=M2, version=M3 or version=M4 for micro format symbols. If unspecified, the encoder will select the version of the symbol that is the minimum size to represent the given data at the selected error correction level. The format option is used to select between format=full and format=micro symbol types. By default, full format symbols will be generated. The raw option denotes that the data field is providing the input as a pre-encoded bitstream (excluding the terminator bits) suitable for direct low-level encoding.

\begin{pspicture}(1in,1in)
\psbarcode{http://www.dante.de}{qrcode}
\end{pspicture}
\begin{pspicture}(1in,1in)
\psbarcode{http://www.dante.de}{eclevel=M}{qrcode}
\end{pspicture}

\begin{pspicture}(1in,1in)
\psbarcode{QR ^067ode}{parse}{qrcode}
\end{pspicture}

\begin{pspicture}(2in,2in)
\psbarcode{QR CODE 1234}{version=10 elevel=Q}{qrcode}
\end{pspicture}

\begin{pspicture}(0.5in,0.5in)
\psbarcode{01234567}{format=micr}{qrcode}
\end{pspicture}
This commentary assumes familiarity with the PostScript language.\footnote{The PostScript Language Tutorial and Cookbook (a.k.a. the Blue Book), which is freely available online, serves as both a useful tutorial and reference manual to the language.}

The code is split cleanly into two types of procedure:

*The encoders* Each of these represents a barcode symbology, e.g. EAN-13 or Code-128. It takes a string containing the barcode data and a string containing a list of options that modify the output of the encoder. It generates a structured representation of the barcode and its text for the symbology, including the calculation of check digits where necessary.

*The renderer* This takes the output of an encoder and generates a visual representation of the barcode.

This means that all barcodes can be generated simply in a similar manner:

\begin{tabular}{|c|}
\hline
(78858101497) (includetext height=0.6) upca barcode \\
(0123456789) (includecheck) interleaved2of5 barcode \\
\hline
\end{tabular}

### 5.1 The Barcode Data Structure

The following table describes the structured representation of a barcode that is passed by an encoder to the renderer as a dictionary when the PostScript is executed.

\begin{tabular}{|c|}
\hline
\end{tabular}
### 5.2 An Encoder

The procedure labelled code2of5 is a simple example of an encoder, which we will now consider. Its purpose is to accept as input a string containing the barcode contents and a string containing a list of options, and to process these in a way that is specific to this encoder, and finally to output an instance of the dictionary-based data structure described in section 5.1 that represents the barcode contents in the Code 2 of 5 symbology.

As with all of the encoders, the input string is assumed to be valid for the corresponding symbology, otherwise the behaviour is undefined.

The variables that we use in this procedure are confined to local scope by declaring the procedure as follows:

```latex
/code2of5 { 
   0 begin
   ...
   end
} bind def
/code2of5 load 0 1 dict put
```

We start by immediately reading the contents strings that are passed as arguments to this procedure by the user.

```latex
/options exch def
/barcode exch def
```

We initialise a few default variables. Those variables corresponding to options that can be enabled with the options argument are initially set to false.

```latex
/includetext false def
/textfont /Courier def
/textsize 10 def
```
The options string is tokenised with each successive token defining either a name value pair which we instantiate or a lone variable that we define as true, allowing us to override the given default variables given above.

```latex
options {
  token false eq {exit} if dup length string cvs (=) search
  true eq {cvlit exch pop exch def} {cvlit true def} ifelse
} loop
```

Since any user given options create variables that are strings we need to convert them back to their intended types.

```latex
/textfont textfont cvlit def
/textsize textsize cvr def
/textpos textpos cvr def
/height height cvr def
```

We then create an array of string encodings for each of the available characters which we then declare in another string. This information can be derived from careful reading of the relevant specification, although this is often surprisingly difficult to obtain.

```latex
/encs
  [ (1111313111) (3111111131) (1131111131) (3131111111)
    (1111311131) (3111311111) (1131311111) (1111113131)
    (3111113111) (1131113111) (313111) (311131)
  ] def
/barchars (0123456789) def
```

We now store the length of the content string and calculate the total number of bars and spaces in the resulting barcode. We initialise a string of size dependant on this length into which we will build the space bar succession. Similarly, we create an array into which we will add the human readable text information.

```latex
/barlen barcode length def
/sbs barlen 10 mul 12 add string def
/txt barlen array def
```

We now begin to populate the space bar succession by adding the encoding of the start character to the beginning.

```latex
sbs 0 encs 10 get putinterval
```

We now enter the main loop which iterates over the content string from start to finish, looking up the encoding for each character, adding this to the space bar succession.

It is important to understand how the encoding for a given character is derived. Firstly, given a character, we find its position in the string of all available characters. We then use this position to index the array of character encodings to obtain the encoding for the given character, which is added to the space/bar succession. Likewise, the character is added to the array of human readable text along with positioning and font information.
5.3 The Renderer

The procedure labelled barcode is known as the renderer, which we now consider. Its purpose is to accept as input an instance of the dictionary-based data structure described in section 5.1 that represents a barcode in some arbitrary symbology and produce a visual rendering of this at the current point.

The variables that we use in this procedure are confined to local scope by declaring the procedure as follows:

```
/barcode {
    0 begin
    ...
    end
} bind def
```

We then immediately read the dictionary-based data structure which is passed as a single argument to this procedure by an encoder, from which we extract the space bar succession, bar height succession and bar base succession.
We attempt to extract from the dictionary the array containing the information about human readable text. However, this may not exist in the dictionary in which case we create a default empty array.

We have extracted or derived all of the necessary information from the input, and now use the space bar succession, bar height succession and bar base succession in calculations that create a single array containing elements that give coordinates for each of the bars in the barcode.

We start by creating a bars array that is half the length of the space bar succession. We build this by repeatedly adding array elements that contain the height, x-coordinate, y-coordinate and width of single bars. The height and y-coordinates are read from the bar height succession and the bar base succession, respectively, whilst the x-coordinate and the width are made from a calculation of the total indent, based on the space bar succession and a compensating factor that accounts for ink spread.

Finally, we perform the actual rendering in two phases. Firstly we use the contents of the bars array that we just built to render each of the bars, and secondly we use the contents of the text array extracted from the input argument to render the text. We make an efficiency saving here by not performing loading and rescaling of a font if the scale factor for the font size is 0. The graphics state is preserved across calls to this procedure to prevent unexpected interference with the users environment.
5.4 Notes Regarding Coding Style

PostScript programming veterans are encouraged to remember that the majority of people who read the code are likely to have little, if any, prior knowledge of the language. To encourage development, the code has been written with these goals in mind:

- That it be easy to use and to comprehend
- That it be easy to modify and enhance

To this end the following points should be observed for all new code submissions:

- New encoders should be based on the code of a similar existing encoder
- Include comments where these clarify the operations involved, particular where something unexpected happens
- Prefer simplicity to efficiency and clarity to obfuscation, except where this will be a problem

5.5 Installing the Barcode Generation Capability into a Printer’s Virtual Machine

Most genuine PostScript printers allow procedures to be defined such that they persist across different jobs through the use of the `exitserver` command. If your printer supports this then you will be able to print the main code containing the definitions of all the encoders and the renderer once, e.g. soon after the device is turned on, and later omit these definitions from each of the barcode documents that you print.

To install the barcode generation capabilities into the virtual machine of a PostScript printer you need to uncomment a line near the top of the code so that it reads:

```pseudocode
serverdict begin 0 exitserver
```

Once this code is printed the procedural definitions for the encoders and the renderer will remain defined across all jobs until the device is reset either by power-cycling or with the following code:

```pseudocode
serverdict begin 0 exitserver systemdict /quit get exec
```
6 QR code in the margin

The margin notes are placed with the help of the macro of the same name \marginnote, which itself needs package-marginnote. It is loaded by default if the macro doesn’t exist.

\begin{verbatim}
\marginnote{\href{http://mirror.ctan.org/help/Catalogue/bytopic.html#barcode}{\LaTeX\textcopyright\hspace{0.1em}2004-2008, LaTeX3 Team}\vphantom{a}}
\end{verbatim}

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.
information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

### 7 List of all optional arguments for \texttt{pst-barcode}

<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>fontstyle</td>
<td>ordinary</td>
<td>\texttt{\small}</td>
</tr>
<tr>
<td>transx</td>
<td>ordinary</td>
<td>0</td>
</tr>
<tr>
<td>transy</td>
<td>ordinary</td>
<td>0</td>
</tr>
<tr>
<td>scalex</td>
<td>ordinary</td>
<td>1</td>
</tr>
<tr>
<td>scaley</td>
<td>ordinary</td>
<td>1</td>
</tr>
<tr>
<td>rotate</td>
<td>ordinary</td>
<td>0</td>
</tr>
<tr>
<td>file</td>
<td>boolean</td>
<td>true</td>
</tr>
</tbody>
</table>

### References


Index

borderbottom, 5
borderleft, 5
borderright, 5
bordertop, 5
borderwidth, 5

code39, 7, 8

ean13, 8
ean2, 10
ean5, 10
ean8, 9

Environment
   pspicture, 4, 7

\fbox, 7
file, 4
font, 5

height, 5

includecheck, 5
includecheckintext, 5
includetext, 5
inksread, 5
isbn, 11

Keyword
   file, 4
   rotate, 4
   scalex, 4
   scaley, 4
   transx, 4
   transy, 4

Macro
   \fbox, 7
   \makebox, 4
   \marginnote, 30
   \parbox, 7
   \psbarcode, 4
   \makebox, 4
   \marginnote, 30
   \marginnote, 30

Package
   narginnote, 30
   pstricks, 4

\parbox, 7
parse, 5

PostScript
   borderbottom, 5
   borderleft, 5
   borderright, 5
   bordertop, 5
   borderwidth, 5
code39, 7, 8
ean13, 8
ean2, 10
ean5, 10
ean8, 9
font, 5
height, 5
includecheck, 5
includecheckintext, 5
includetext, 5
inksread, 5
isbn, 11
parse, 5
showborder, 5
textpos, 5
textsize, 5
upca, 9
upce, 10
width, 5
\psbarcode, 4
pspicture, 4, 7
pstricks, 4

rotate, 4
scalex, 4
scaley, 4
showborder, 5

\textpos, 5
\textsize, 5
transx, 4
transy, 4
upca, 9
upce, 10
width, 5