

GB/T 7714 BibT_EX style

Zeping Lee*

2021/12/08 v2.1.3

摘要

The gbt7714 package provides a BibT_EX implementation for the China's national bibliography style standard GB/T 7714. It consists of .bst files for numeric and author-date styles as well as a L^AT_EX package which provides the citation style defined in the standard. It is compatible with natbib and supports language detection (Chinese and English) for each bibliography entry.

1 简介

GB/T 7714—2015 《信息与文献 参考文献著录规则》^[1]（以下简称“国标”）是中国的参考文献格式推荐标准。国内的绝大部分学术期刊、学位论文都使用了基于该标准的格式。本宏包是国标的 BibT_EX^[2] 实现，具有以下特性：

- 兼容 natbib 宏包^[3]。
- 支持“顺序编码制”和“著者-出版年制”两种风格。
- 自动识别语言并进行相应处理。
- 提供了简单的接口供用户修改样式。
- 同时提供了 2005 版的 .bst 文件。

本宏包的主页：<https://github.com/zepinglee/gbt7714-bibtex-style>。

2 版本 v2.0 的重要修改

从 v2.0 版本开始(2020-03-04),用户必须在文档中使用 \bibliographystyle 命令选择参考文献样式,如 gbt7714-numerical 或 gbt7714-author-year。在早期的版本中,选择文献样式的方法是将 numbers 或 super 等参数传递给 gbt7714,而不能使用 \bibliographystyle。这跟标准的 LaTeX 接口不一致,所以将被弃用。

*zepinglee AT gmail.com

3 使用方法

以下是 gbt7714 宏包的一个简单示例。

```
\documentclass{ctexart}
\usepackage{gbt7714}
\bibliographystyle{gbt7714-numerical}
\begin{document}
  \cite{...}
  ...
  \bibliography{bibfile}
\end{document}
```

按照国标的规定，参考文献的标注体系分为“顺序编码制”和“著者-出版年制”。用户应在导言区调用宏包 gbt7714，并且使用 \bibliographystyle 命令选择参考文献表的样式，比如：

```
\bibliographystyle{gbt7714-numerical} % 顺序编码制
```

或者

```
\bibliographystyle{gbt7714-author-year} % 著者-出版年制
```

此外还可以使用 2005 版的格式 gbt7714-2005-numerical 和 gbt7714-2005-numerical。

注意，版本 v2.0 更改了设置参考文献表样式的方法，要求直接使用 \bibliographystyle，不再使用宏包的参数，而且更改了 bst 的文件名。

顺序编码制的引用标注默认使用角标式，如“张三^[2]提出”。如果要使用正文模式，如“文献 [3] 中说明”，可以使用 \citestyle 命令进行切换：

```
\citestyle{numbers}
```

同一处引用多篇文献时，应当将各篇文献的 key 一同写在 \cite 命令中。如遇连续编号，默认会自动转为起讫序号并用短横线连接（见 natbib 的 compress 选项）。如果要对引用的编号进行自动排序，需要在调用 gbt7714 时加 sort&compress 参数：

```
\usepackage[sort&compress]{gbt7714}
```

这些参数会传给 natbib 处理。

若需要标出引文的页码，可以标在 \cite 的可选参数中，如 \cite[42]{knuth84}。更多的引用标注方法可以参考 natbib 宏包的使用说明^[3]。

使用时需要注意以下几点：

- `.bib` 数据库应使用 UTF-8 编码。
- 使用著者-出版年制参考文献表时，中文的文献必须在 `key` 域填写作者姓名的拼音，才能按照拼音排序，详见第 6 节。

4 文献类型

国标中规定了 16 种参考文献类型，表 1 列举了 `bib` 数据库中对应的文献类型。这些尽可能兼容 `BibTEX` 的标准类型，但是新增了若干文献类型（带 * 号）。

表 1: 全部文献类型

文献类型	标识代码	Entry Type
普通图书	M	book
图书的析出文献	M	incollection
会议录	C	proceedings
会议录的析出文献	C	inproceedings 或 conference
汇编	G	collection*
报纸	N	newspaper*
期刊的析出文献	J	article
学位论文	D	mastersthesis 或 phdthesis
报告	R	techreport
标准	S	standard*
专利	P	patent*
数据库	DB	database*
计算机程序	CP	software*
电子公告	EB	online*
档案	A	archive*
舆图	CM	map*
数据集	DS	dataset*
其他	Z	misc

5 著录项目

由于国标中规定的著录项目多于 `BibTEX` 的标准域，必须新增一些著录项目（带 * 号），这些新增的类型在设计时参考了 `BibLaTEX`，如 `date` 和 `urldate`。本宏包支持的全部域如下：

author 主要责任者

title 题名
mark* 文献类型标识
medium* 载体类型标识
translator* 译者
editor 编辑
organization 组织（用于会议）
booktitle 图书题名
series 系列
journal 期刊题名
edition 版本
address 出版地
publisher 出版者
school 学校（用于 phdthesis）
institution 机构（用于 techreport）
year 出版年
volume 卷
number 期（或者专利号）
pages 引文页码
date* 更新或修改日期
urldate* 引用日期
url 获取和访问路径
doi 数字对象唯一标识符
langid* 语言
key 拼音（用于排序）

不支持的 BibT_EX 标准著录项目有 `annotate`, `chapter`, `crossref`, `month`, `type`。

本宏包默认情况下可以自动识别文献语言，并自动处理文献类型和载体类型标识，但是在少数情况下需要用户手动指定，如：

```

@misc{citekey,
  langid = {japanese},
  mark   = {Z},
  medium = {DK},
  ...
}
  
```

可选的语言有 `english`, `chinese`, `japanese`, `russian`。

6 文献列表的排序

国标规定参考文献表采用著者-出版年制组织时，各篇文献首先按文种集中，然后按著者字顺和出版年排列；中文文献可以按著者汉语拼音字顺排列，也可以按著者的笔画笔顺排列。然而由于 BibT_EX 功能的局限性，无法自动获取著者姓名的拼音或笔画笔顺，所以必须在 bib 数据库中的 key 域手动录入著者姓名的拼音，如：

```
@book{capital,
  author = {马克思 and 恩格斯},
  key    = {ma3 ke4 si1 & en1 ge2 si1},
  ...
}
```

7 自定义样式

BibT_EX 对自定义样式的支持比较有限，所以用户只能通过修改 bst 文件来修改文献列表的格式。本宏包提供了一些接口供用户更方便地修改。

在 bst 文件开始处的 load.config 函数中，有一组配置参数用来控制样式，表 2 列出了每一项的默认值和功能。若变量被设为 #1 则表示该项被启用，设为 #0 则不启用。默认的值是严格遵循国标的配置。

若用户需要定制更多内容，可以学习 bst 文件的语法并修改^[4-6]，或者联系作者。

8 相关工作

TeX 社区也有其他关于 GB/T 7714 系列参考文献标准的工作。2005 年吴凯^[7]发布了基于 GB/T 7714—2005 的 BibT_EX 样式，支持顺序编码制和著者出版年制两种风格。李志奇^[8]发布了严格遵循 GB/T 7714—2005 的 BibLaTeX 的样式。胡海星^[9]提供了另一个 BibT_EX 实现，还给每行 bst 代码写了 java 语言注释。沈周^[10]基于 biblatex-casperiector^[11] 进行修改，以符合国标的格式。胡振震发布了符合 GB/T 7714—2015 标准的 BibLaTeX 参考文献样式^[12]，并进行了比较完善的持续维护。

表 2: 参考文献表样式的配置参数

参数值	默认值	功能
uppercase.name	#1	将著者姓名转为大写
max.num.authors	#3	输出著者的最多数量
year.after.author	#0	年份置于著者之后
period.after.author	#0	著者和年份之间使用句点连接
italic.book.title	#0	西文书籍名使用斜体
sentence.case.title	#1	将西文的题名转为 sentence case
link.title	#0	在题名上添加 url 的超链接
title.in.journal	#1	期刊是否显示标题
show.patent.country	#0	专利题名是否含国别
space.before.mark	#0	文献类型标识前是否有空格
show.mark	#1	显示文献类型标识
show.medium.type	#1	显示载体类型标识
italic.journal	#0	西文期刊名使用斜体
show.missing.address.publisher	#0	出版项缺失时显示“出版者不详”
space.before.pages	#1	页码与前面的冒号之间有空格
only.start.page	#0	只显示起始页码
wave.dash.in.pages	#0	起止页码使用波浪号
show.urldate	#1	显示引用日期 urldate
show.url	#1	显示 url
show.doi	#1	显示 DOI
show.preprint	#1	显示预印本信息
show.note	#0	显示 note 域的信息
end.with.period	#1	结尾加句点

参考文献

- [1] 中国标准化委员会. 信息与文献 参考文献著录规则: GB/T 7714—2015[S]. 北京: 中国标准出版社, 2015.
- [2] PATASHNIK O. BibT_EXing[M/OL]. 1988. <http://mirrors.ctan.org/biblio/bibtex/base/btxdoc.pdf>.
- [3] DALY P W. Natural sciences citations and references[M/OL]. 1999. <http://mirrors.ctan.org/macros/latex/contrib/natbib/natbib.pdf>.
- [4] PATASHNIK O. Designing BibT_EX styles[M/OL]. 1988. <http://mirrors.ctan.org/biblio/bibtex/base/btxhak.pdf>.

- [5] MARKEY N. Tame the beast[M/OL]. 2003. http://mirrors.ctan.org/info/bibtex/tamethebeast/ttb_en.pdf.
- [6] MITTELBAACH F, GOOSSENS M, BRAAMS J, et al. The L^AT_EX companion[M]. 2nd ed. Reading, MA, USA: Addison-Wesley, 2004.
- [7] 吴凯. 发布 GBT7714-2005.bst version1 Beta 版 [EB/OL]. 2006. CTeX 论坛 (已关闭) .
- [8] 李志奇. 基于 biblatex 的符合 GB/T7714—2005 的中文文献生成工具 [EB/OL]. 2013. CTeX 论坛 (已关闭) .
- [9] 胡海星. A GB/T 7714—2005 national standard compliant BibTeX style[EB/OL]. 2013. <https://github.com/Haixing-Hu/GBT7714-2005-BibTeX-Style>.
- [10] 沈周. 基于 caspervector 改写的符合 GB/T 7714—2005 标准的参考文献格式 [EB/OL]. 2016. <https://github.com/szsdk/biblatex-gbt77142005>.
- [11] VECTOR C T. biblatex 参考文献和引用样式: caspervector[M/OL]. 2012. <http://mirrors.ctan.org/macros/latex/contrib/biblatex-contrib/biblatex-caspervector/doc/caspervector.pdf>.
- [12] 胡振震. 符合 GB/T 7714—2015 标准的 biblatex 参考文献样式 [M/OL]. 2016. <http://mirrors.ctan.org/macros/latex/contrib/biblatex-contrib/biblatex-gb7714-2015/biblatex-gb7714-2015.pdf>.

A 宏包的代码实现

兼容过时的接口

```
1 (*package)
2 \newif\ifgbt@legacy@interface
3 \newif\ifgbt@mmxv
4 \newif\ifgbt@numerical
5 \newif\ifgbt@super
6 \newcommand\gbt@obsolete@option[1]{%
7   \PackageWarning{gbt7714}{The option "#1" is obsolete}%
8 }
9 \DeclareOption{2015}{%
10   \gbt@obsolete@option{2015}%
11   \gbt@legacy@interfacetrue
12   \gbt@mmxvtrue
13 }
14 \DeclareOption{2005}{%
15   \gbt@obsolete@option{2005}%
16   \gbt@legacy@interfacetrue
17   \gbt@mmxvfalse
18 }
19 \DeclareOption{super}{%
20   \gbt@obsolete@option{super}%
21   \gbt@legacy@interfacetrue
22   \gbt@numericaltrue
23   \gbt@supertrue
24 }
25 \DeclareOption{numbers}{%
26   \gbt@obsolete@option{numbers}%
27   \gbt@legacy@interfacetrue
28   \gbt@numericaltrue
29   \gbt@superfalse
30 }
31 \DeclareOption{authoryear}{%
32   \gbt@obsolete@option{authoryear}%
33   \gbt@legacy@interfacetrue
34   \gbt@numericalfalse
35 }
```

将选项传递给 natbib

```
36 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{natbib}}
37 \ProcessOptions\relax
```


调用宏包，注意只需要 `compress` 不需要 `sort`。

```
38 \RequirePackage{natbib}
```

```
39 \RequirePackage{url}
```

如果将 `compress` 传给 `natbib` 容易导致 `option clash`。这里直接修改内部命令。

```
40 \def\NAT@cmprs{\@ne}
```

`\citestyle` 定义接口切换引用文献的标注法,可用 `\citestyle` 调用 `numerical` 或 `authoryear`, 参见 `natbib`。

```
41 \renewcommand\newblock{\space}
```

```
42 \newcommand\bibstyle@super{\bibpunct{[}{]}{,}{s}{,}{\textsuperscript{,}}}
```

```
43 \newcommand\bibstyle@numbers{\bibpunct{[}{]}{,}{n}{,}{,}}
```

```
44 \newcommand\bibstyle@authoryear{\bibpunct{({}{})}{;}{a}{,}{,}}
```

```
45 \newcommand\bibstyle@inline{\bibstyle@numbers}
```

在使用 `\bibliographystyle` 时自动切换引用文献的标注的样式。

```
46 \@namedef{bibstyle@gbt7714-numerical}{\bibstyle@super}
```

```
47 \@namedef{bibstyle@gbt7714-author-year}{\bibstyle@authoryear}
```

```
48 \@namedef{bibstyle@gbt7714-2005-numerical}{\bibstyle@super}
```

```
49 \@namedef{bibstyle@gbt7714-2005-author-year}{\bibstyle@authoryear}
```

`\cite` 下面修改 `natbib` 的引用格式。为了减少依赖的宏包，这里直接重定义命令不使用 `\patchcmd`。

`Super` 样式的 `\citep` 的页码也为上标。另外加上 `\kern\p@` 去掉上标式引用后与中文之间多余的空格，参考 [tuna/thuthesis#624](#)。

```
50 \renewcommand\NAT@citesuper[3]{%
```

```
51 \if*#2*\else
```

```
52 \if*#2*\else
```

```
53 #2\NAT@spacechar
```

```
54 \fi
```

```
55 % \unskip\kern\p@\textsuperscript{\NAT@open#1\NAT@close}%
```

```
56 % \if*#3*\else\NAT@spacechar#3\fi\else #1\fi\endgroup}
```

```
57 \unskip\kern\p@
```

```
58 \textsuperscript{%
```

```
59 \NAT@open
```

```
60 #1%
```

```
61 \NAT@close
```

```
62 \if*#3*\else
```

```
63 #3%
```

```
64 \fi
```

```
65 }%
```

```

66 \kern\p@
67 \else
68 #1%
69 \fi
70 \endgroup
71 }

```

将 numbers 样式的 \citep 的页码置于括号外。

```

72 \renewcommand\NAT@citenum[3]{%
73 \ifNAT@swa
74 \NAT@@open
75 \if*#2*\else
76 #2\NAT@spacechar
77 \fi
78 % #1\if*#3*\else\NAT@cmt#3\fi\NAT@close\else#1\fi\endgroup}
79 #1\NAT@close
80 \if*#3*\else
81 \textsuperscript{#3}%
82 \fi
83 \else
84 #1%
85 \fi
86 \endgroup
87 }

```

Numerical 模式的 \citet 的页码：

```

88 \def\NAT@citexnum[#1][#2]#3{%
89 \NAT@reset@parser
90 \NAT@sort@cites{#3}%
91 \NAT@reset@citea
92 \@cite{\def\NAT@num{-1}\let\NAT@last@yr\relax\let\NAT@nm\@empty
93 \@for\@citeb:=\NAT@cite@list\do
94 {\@safe@activetrue
95 \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}%
96 \@safe@activesfalse
97 \@ifundefined{b@\@citeb\@extra@b@citeb}{%
98 {\reset@font\bfseries?}
99 \NAT@citeundefined\PackageWarning{natbib}%
100 {Citation '\@citeb' on page \thepage \space undefined}}%
101 {\let\NAT@last@num\NAT@num\let\NAT@last@nm\NAT@nm
102 \NAT@parse{\@citeb}%
103 \ifNAT@longnames\@ifundefined{bv@\@citeb\@extra@b@citeb}{%
104 \let\NAT@name=\NAT@all@names

```

```

105     \global\@namedef{bv@\@citeb\@extra@b@citeb}{}}{}%
106     \fi
107     \ifNAT@full\let\NAT@nm\NAT@all@names\else
108         \let\NAT@nm\NAT@name\fi
109     \ifNAT@swa
110         \@ifnum{\NAT@ctype>\@ne}{%
111             \citea
112             \NAT@hyper@{\@ifnum{\NAT@ctype=\tw@}{\NAT@test{\NAT@ctype}}{\NAT@alias}}%
113         }{%
114             \@ifnum{\NAT@cmprs>\z@}{%
115                 \NAT@ifcat@num\NAT@num
116                 {\let\NAT@nm=\NAT@num}%
117                 {\def\NAT@nm{-2}}%
118                 \NAT@ifcat@num\NAT@last@num
119                 {\@tempcnta=\NAT@last@num\relax}%
120                 {\@tempcnta\m@ne}%
121                 \@ifnum{\NAT@nm=\@tempcnta}{%
122                     \@ifnum{\NAT@merge>\@ne}{\NAT@last@yr@mbbox}%
123                 }{%
124                     \advance\@tempcnta by\@ne
125                     \@ifnum{\NAT@nm=\@tempcnta}{%

```

在顺序编码制下，**natbib** 只有在三个以上连续文献引用才会使用连接号，这里修改为允许两个引用使用连接号。

```

126         % \ifx\NAT@last@yr\relax
127         %     \def@NAT@last@yr{\@citea}%
128         % \else
129         %     \def@NAT@last@yr{--\NAT@penalty}%
130         % \fi
131         \def@NAT@last@yr{-\NAT@penalty}%
132     }{%
133         \NAT@last@yr@mbbox
134     }%
135 }%
136 }{%
137     \@tempswatrue
138     \@ifnum{\NAT@merge>\@ne}{\@ifnum{\NAT@last@num=\NAT@num\relax}{\@tempswafalse}{}}{}%
139     \if@tempswa\NAT@citea@mbbox\fi
140 }%
141 }%
142 \NAT@def@citea
143 \else
144     \ifcase\NAT@ctype

```

```

145 \ifx\NAT@last@nm\NAT@nm \NAT@yrsep\NAT@penalty\NAT@space\else
146 \@citea \NAT@test{\@ne}\NAT@spacechar\NAT@mbox{\NAT@super@kern\NAT@@open}%
147 \fi
148 \if*#1*\else#1\NAT@spacechar\fi
149 \NAT@mbox{\NAT@hyper@{\@citenumfont{\NAT@num}}}%
150 \NAT@def@citea@box
151 \or
152 \NAT@hyper@citea@space{\NAT@test{\NAT@ctype}}%
153 \or
154 \NAT@hyper@citea@space{\NAT@test{\NAT@ctype}}%
155 \or
156 \NAT@hyper@citea@space\NAT@alias
157 \fi
158 \fi
159 }%
160 }%
161 \@ifnum{\NAT@cmprs>\z@}{\NAT@last@yr}{}%
162 \ifNAT@swa\else

```

将页码放在括号外边，并且置于上标。

```

163 % \@ifnum{\NAT@ctype=\z@}{%
164 % \if*#2*\else\NAT@cmt#2\fi
165 % }{%
166 \NAT@mbox{\NAT@close}%
167 \@ifnum{\NAT@ctype=\z@}{%
168 \if*#2*\else
169 \textsuperscript{#2}%
170 \fi
171 }{%
172 \NAT@super@kern
173 \fi
174 }{#1}{#2}%
175 }%

```

Author-year 模式的 \citep 的页码：

```

176 \renewcommand\NAT@cite%
177 [3]{\ifNAT@swa\NAT@@open\if*#2*\else#2\NAT@spacechar\fi
178 #1\NAT@close\if*#3*\else\textsuperscript{#3}\fi\else#1\fi\endgroup}

```

Author-year 模式的 \citet 的页码：

```

179 \def\NAT@citex%
180 [#1][#2]#3{%
181 \NAT@reset@parser

```

```

182 \NAT@sort@cites{#3}%
183 \NAT@reset@citea
184 \@cite{\let\NAT@nm\@empty\let\NAT@year\@empty
185 \@for\@citeb:=\NAT@cite@list\do
186 {\@safe@activestrue
187 \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}%
188 \@safe@activesfalse
189 \@ifundefined{b@\@citeb\@extra@b@citeb}{\@citea%
190 {\reset@font\bfseries ?}\NAT@citeundefined
191 \PackageWarning{natbib}%
192 {Citation '\@citeb' on page \thepage \space undefined}\def\NAT@date{}}}%
193 {\let\NAT@last@nm=\NAT@nm\let\NAT@last@yr=\NAT@year
194 \NAT@parse{\@citeb}%
195 \ifNAT@longnames\@ifundefined{bv@\@citeb\@extra@b@citeb}{%
196 \let\NAT@name=\NAT@all@names
197 \global\@namedef{bv@\@citeb\@extra@b@citeb}{}}{}%
198 \fi
199 \ifNAT@full\let\NAT@nm\NAT@all@names\else
200 \let\NAT@nm\NAT@name\fi
201 \ifNAT@swa\ifcase\NAT@ctype
202 \if\relax\NAT@date\relax
203 \@citea\NAT@hyper@{\NAT@nmfmt{\NAT@nm}\NAT@date}%
204 \else
205 \ifx\NAT@last@nm\NAT@nm\NAT@yrsep
206 \ifx\NAT@last@yr\NAT@year
207 \def\NAT@temp{?}%
208 \ifx\NAT@temp\NAT@exlab\PackageWarningNoLine{natbib}%
209 {Multiple citation on page \thepage: same authors and
210 year\MessageBreak without distinguishing extra
211 letter,\MessageBreak appears as question mark}\fi
212 \NAT@hyper@{\NAT@exlab}%
213 \else\unskip\NAT@spacechar
214 \NAT@hyper@{\NAT@date}%
215 \fi
216 \else
217 \@citea\NAT@hyper@{%
218 \NAT@nmfmt{\NAT@nm}%
219 \hyper@natlinkbreak{%
220 \NAT@aysep\NAT@spacechar}{\@citeb\@extra@b@citeb
221 }%
222 \NAT@date
223 }%

```

```

224     \fi
225   \fi
226   \or\@citea\NAT@hyper@{\NAT@nmfmt{\NAT@nm}}%
227   \or\@citea\NAT@hyper@{\NAT@date}%
228   \or\@citea\NAT@hyper@{\NAT@alias}%
229   \fi \NAT@def@citea
230   \else
231     \ifcase\NAT@ctype
232     \if\relax\NAT@date\relax
233       \@citea\NAT@hyper@{\NAT@nmfmt{\NAT@nm}}%
234     \else
235       \ifx\NAT@last@nm\NAT@nm\NAT@yrsep
236         \ifx\NAT@last@yr\NAT@year
237           \def\NAT@temp{{?}}%
238           \ifx\NAT@temp\NAT@exlab\PackageWarningNoLine{natbib}%
239             {Multiple citation on page \thepage: same authors and
240             year\MessageBreak without distinguishing extra
241             letter,\MessageBreak appears as question mark}\fi
242           \NAT@hyper@{\NAT@exlab}%
243         \else
244           \unskip\NAT@spacechar
245           \NAT@hyper@{\NAT@date}%
246         \fi
247       \else
248         \@citea\NAT@hyper@{%
249           \NAT@nmfmt{\NAT@nm}%
250           \hyper@natlinkbreak{\NAT@spacechar\NAT@@open\if*#1*\else#1\NAT@spacechar\fi}%
251           {\@citeb\@extra@b@citeb}%
252           \NAT@date
253         }%
254       \fi
255     \fi
256     \or\@citea\NAT@hyper@{\NAT@nmfmt{\NAT@nm}}%
257     \or\@citea\NAT@hyper@{\NAT@date}%
258     \or\@citea\NAT@hyper@{\NAT@alias}%
259     \fi
260     \if\relax\NAT@date\relax
261       \NAT@def@citea
262     \else
263       \NAT@def@citea@close
264     \fi
265   \fi

```

```
266 }}\ifNAT@swa\else
```

将页码放在括号外边，并且置于上标。

```
267 % \if*#2*\else\NAT@cmt#2\fi
268 \if\relax\NAT@date\relax\else\NAT@@close\fi
269 \if*#2*\else\textsuperscript{#2}\fi
270 \fi}{#1}{#2}}
```

thebibliography 参考文献列表的标签左对齐

```
271 \renewcommand\@biblabel[1]{#1\hfill}
```

\url 使用 xurl 宏包的方法，增加 URL 可断行的位置。

```
272 \g@addto@macro\UrlBreaks{%
273 \do0\do1\do2\do3\do4\do5\do6\do7\do8\do9%
274 \doA\doB\doC\doD\doE\doF\doG\doH\doI\doJ\doK\doL\doM
275 \doN\doO\doP\doQ\doR\doS\doT\doU\doV\doW\doX\doY\doZ
276 \doa\dob\doc\do d\doe\do f\dog\do h\do i\do j\do k\do l\do m
277 \do n\do o\do p\do q\do r\do s\do t\do u\do v\do w\do x\do y\do z
278 }
279 \Urlmuskip=0mu plus 0.1mu
```

兼容 v2.0 前过时的接口：

```
280 \newif\ifgibt@bib@style@written
281 \@ifpackageloaded{chapterbib}{}%
282 \def\bibliography#1{%
283 \ifgibt@bib@style@written\else
284 \bibliographystyle{gbt7714-numerical}%
285 \fi
286 \if@filesw
287 \immediate\write\@auxout{\string\bibdata{\zap@space#1 \@empty}}%
288 \fi
289 \@input@{\jobname.bbl}}
290 \def\bibliographystyle#1{%
291 \gibt@bib@style@writtenttrue
292 \ifx\@begindocumenthook\@undefined\else
293 \expandafter\AtBeginDocument
294 \fi
295 {\if@filesw
296 \immediate\write\@auxout{\string\bibstyle{#1}}%
297 \fi}%
298 }%
299 }
300 \ifgibt@legacy@interface
```

```

301 \ifgbt@numerical
302   \ifgbt@super\else
303     \citestyle{numbers}
304   \fi
305   \bibliographystyle{gbt7714-numerical}
306 \else
307   \bibliographystyle{gbt7714-author-year}
308 \fi
309 \fi
310 </package>

```

B BibTeX 样式的代码实现

B.1 自定义选项

bst 这里定义了一些变量用于定制样式，可以在下面的 `load.config` 函数中选择是否启用。

```

311 (*author-year | numerical)
312 INTEGERS {
313   citation.et.al.min
314   citation.et.al.use.first
315   bibliography.et.al.min
316   bibliography.et.al.use.first
317   uppercase.name
318   terms.in.macro
319   year.after.author
320   period.after.author
321   italic.book.title
322   sentence.case.title
323   link.title
324   title.in.journal
325   show.patent.country
326   show.mark
327   space.before.mark
328   show.medium.type
329   slash.for.extraction
330   in.booktitle
331   short.journal
332   italic.journal
333   bold.journal.volume
334   show.missing.address.publisher
335   space.before.pages
336   only.start.page
337   wave.dash.in.pages
338   show.urldate
339   show.url
340   show.doi
341   show.preprint
342   show.note

```



```

343 show.english.translation
344 end.with.period
345 (*author-year)
346 lang.zh.order
347 lang.ja.order
348 lang.en.order
349 lang.ru.order
350 lang.other.order
351 </author-year>
352 }
353

```

下面每个变量若被设为 **#1** 则启用该项，若被设为 **#0** 则不启用。默认的值是严格遵循国标的配置。

```

354 FUNCTION {load.config}
355 {

```

如果姓名的数量大于等于 **et.al.min**，只著录前 **et.al.use.first** 个，其后加“et al.”或“等”。

```

356 (*!ucas)
357 #2 'citation.et.al.min :=
358 #1 'citation.et.al.use.first :=
359 </!ucas)
360 (*ucas)
361 #3 'citation.et.al.min :=
362 #1 'citation.et.al.use.first :=
363 </ucas)
364 #4 'bibliography.et.al.min :=
365 #3 'bibliography.et.al.use.first :=

```

英文姓名转为全大写：

```

366 (*!(no-uppercase | thu))
367 #1 'uppercase.name :=
368 </!(no-uppercase | thu))
369 (*no-uppercase | thu)
370 #0 'uppercase.name :=
371 </no-uppercase | thu)

```

使用 TeX 宏输出“和”、“等”

```

372 (*!(macro | ucas))
373 #0 'terms.in.macro :=
374 </!(macro | ucas))
375 (*macro | ucas)
376 #1 'terms.in.macro :=
377 </macro | ucas)

```

将年份置于著者后面（著者-出版年制默认）

```

378 (*numerical | ucas)
379 #0 'year.after.author :=
380 </numerical | ucas)
381 (*author-year&!ucas)
382 #1 'year.after.author :=
383 </author-year&!ucas)

```

采用著者-出版年制时，作者姓名与年份之间使用句点连接：

```
384 < *numerical >
385 #1 'period.after.author :=
386 < /numerical >
387 < *author-year >
388 < *2015&!(period | ustc) >
389 #0 'period.after.author :=
390 < /2015&!(period | ustc) >
391 < *period | 2005 | ustc >
392 #1 'period.after.author :=
393 < /period | 2005 | ustc >
394 < /author-year >
```

书名使用斜体：

```
395 < *!italic-book-title >
396 #0 'italic.book.title :=
397 < /!italic-book-title >
398 < *italic-book-title >
399 #1 'italic.book.title :=
400 < /italic-book-title >
```

英文标题转为 sentence case（句首字母大写，其余小写）：

```
401 < *!no-sentence-case >
402 #1 'sentence.case.title :=
403 < /!no-sentence-case >
404 < *no-sentence-case >
405 #0 'sentence.case.title :=
406 < /no-sentence-case >
```

在标题添加超链接：

```
407 < *!link-title >
408 #0 'link.title :=
409 < /!link-title >
410 < *link-title >
411 #1 'link.title :=
412 < /link-title >
```

期刊是否含标题：

```
413 < *!no-title-in-journal >
414 #1 'title.in.journal :=
415 < /!no-title-in-journal >
416 < *no-title-in-journal >
417 #0 'title.in.journal :=
418 < /no-title-in-journal >
```

专利题名是否含专利国别

```
419 < *!(show-patent-country | 2005 | ustc | thu) >
420 #0 'show.patent.country :=
421 < /!(show-patent-country | 2005 | ustc | thu) >
422 < *(show-patent-country | 2005 | ustc | thu) >
423 #1 'show.patent.country :=
424 < /(show-patent-country | 2005 | ustc | thu) >
```

著录文献类型标识（比如“[M/OL]”）：

```

425 <!*no-mark>
426 #1 'show.mark :=
427 </!no-mark>
428 <*no-mark>
429 #0 'show.mark :=
430 </no-mark>

```

文献类型标识前是否有空格:

```

431 <!*space-before-mark>
432 #0 'space.before.mark :=
433 </!space-before-mark>
434 <*space-before-mark>
435 #1 'space.before.mark :=
436 </space-before-mark>

```

是否显示载体类型标识 (比如“/OL”):

```

437 <!*no-medium-type>
438 #1 'show.medium.type :=
439 </!no-medium-type>
440 <*no-medium-type>
441 #0 'show.medium.type :=
442 </no-medium-type>

```

使用“//”表示析出文献

```

443 <!*no-slash>
444 #1 'slash.for.extraction :=
445 </!no-slash>
446 <*no-slash>
447 #0 'slash.for.extraction :=
448 </no-slash>

```

使用“In:”表示析出文献

```

449 #0 'in.booktitle :=

```

期刊名使用缩写:

```

450 <!*short-journal>
451 #0 'short.journal :=
452 </!short-journal>
453 <*short-journal>
454 #1 'short.journal :=
455 </short-journal>

```

期刊名使用斜体:

```

456 <!*italic-journal>
457 #0 'italic.journal :=
458 </!italic-journal>
459 <*italic-journal>
460 #1 'italic.journal :=
461 </italic-journal>

```

期刊的卷使用粗体:

```

462 #0 'bold.journal.volume :=

```

无出版地或出版者时, 著录“出版地不详”, “出版者不详”, “S.l.”或“s.n.”:

```

463 <!*sl-sn>
464 #0 'show.missing.address.publisher :=
465 </!sl-sn>
466 <*sl-sn>
467 #1 'show.missing.address.publisher :=
468 </sl-sn>

```

页码与前面的冒号之间是否有空格：

```

469 <!*no-space-before-pages>
470 #1 'space.before.pages :=
471 </!no-space-before-pages>
472 <*no-space-before-pages>
473 #0 'space.before.pages :=
474 </no-space-before-pages>

```

页码是否只含起始页：

```

475 <!*only-start-page>
476 #0 'only.start.page :=
477 </!only-start-page>
478 <*only-start-page>
479 #1 'only.start.page :=
480 </only-start-page>

```

起止页码使用波浪号：

```

481 <!*wave-dash-in-pages>
482 #0 'wave.dash.in.pages :=
483 </!wave-dash-in-pages>
484 <*wave-dash-in-pages>
485 #1 'wave.dash.in.pages :=
486 </wave-dash-in-pages>

```

是否著录非电子文献的引用日期：

```

487 <!*no-urldate>
488 #1 'show.urldate :=
489 </!no-urldate>
490 <*no-urldate>
491 #0 'show.urldate :=
492 </no-urldate>

```

是否著录 URL：

```

493 <!*no-url>
494 #1 'show.url :=
495 </!no-url>
496 <*no-url>
497 #0 'show.url :=
498 </no-url>

```

是否著录 DOI：

```

499 <*(no-doi | 2005)>
500 #1 'show.doi :=
501 </!(no-doi | 2005)>
502 <*no-doi | 2005>
503 #0 'show.doi :=
504 </no-doi | 2005>

```

是否著录 e-print:

```
505 <!*preprint>
506 #1 'show.preprint :=
507 </!preprint>
508 <*preprint>
509 #0 'show.preprint :=
510 </preprint>
```

在每一条文献最后输出注释 (note) 的内容:

```
511 #0 'show.note :=
```

中文文献是否显示英文翻译

```
512 <!*show-english-translation>
513 #0 'show.english.translation :=
514 </!show-english-translation>
515 <*show-english-translation>
516 #1 'show.english.translation :=
517 </show-english-translation>
```

结尾加句点

```
518 <!*no-period-at-end>
519 #1 'end.with.period :=
520 </!no-period-at-end>
521 <*no-period-at-end>
522 #0 'end.with.period :=
523 </no-period-at-end>
```

参考文献表按照“著者-出版年”组织时, 各个文种的顺序:

```
524 <*author-year>
525 #1 'lang.zh.order :=
526 #2 'lang.ja.order :=
527 #3 'lang.en.order :=
528 #4 'lang.ru.order :=
529 #5 'lang.other.order :=
530 </author-year>
531 }
532
```

B.2 The ENTRY declaration

Like Scribe's (according to pages 231-2 of the April '84 edition), but no full-author or editors fields because BibTeX does name handling. The `annotate` field is commented out here because this family doesn't include an annotated bibliography style. And in addition to the fields listed here, BibTeX has a built-in `crossref` field, explained later.

```
533 ENTRY
534 { address
535   archivePrefix
536   author
537   booktitle
538   date
```

```

539     doi
540     edition
541     editor
542     eprint
543     eprinttype
544     howpublished
545     institution
546     journal
547     journaltitle
548     key
549     langid
550     language
551     location
552     mark
553     medium
554     note
555     number
556     organization
557     pages
558     publisher
559     school
560     series
561     shortjournal
562     title
563     translation
564     translator
565     url
566     urldate
567     volume
568     year
569 }
570 { entry.lang entry.is.electronic is.pure.electronic entry.numbered }

```

These string entry variables are used to form the citation label. In a storage pinch, sort.label can be easily computed on the fly.

```

571 { label extra.label sort.label short.label short.list entry.mark entry.url }
572

```

B.3 Entry functions

Each entry function starts by calling `output.bibitem`, to write the `\bibitem` and its arguments to the .BBL file. Then the various fields are formatted and printed by `output` or `output.check`. Those functions handle the writing of separators (commas, periods, `\newblock`'s), taking care not to do so when they are passed a null string. Finally, `fin.entry` is called to add the final period and finish the entry.

A bibliographic reference is formatted into a number of 'blocks': in the open format, a block begins on a new line and subsequent lines of the block are indented. A block may contain more than one sentence (well, not a grammatical sentence, but something to be ended with a sentence ending period). The entry functions should call

new.block whenever a block other than the first is about to be started. They should call new.sentence whenever a new sentence is to be started. The output functions will ensure that if two new.sentence's occur without any non-null string being output between them then there won't be two periods output. Similarly for two successive new.block's.

The output routines don't write their argument immediately. Instead, by convention, that argument is saved on the stack to be output next time (when we'll know what separator needs to come after it). Meanwhile, the output routine has to pop the pending output off the stack, append any needed separator, and write it.

To tell which separator is needed, we maintain an output.state. It will be one of these values: before.all just after the \bibitem mid.sentence in the middle of a sentence: comma needed if more sentence is output after.sentence just after a sentence: period needed after.block just after a block (and sentence): period and \newblock needed. Note: These styles don't use after.sentence

VAR: output.state : INTEGER – state variable for output

The output.nonnull function saves its argument (assumed to be nonnull) on the stack, and writes the old saved value followed by any needed separator. The ordering of the tests is decreasing frequency of occurrence.

由于专著中的析出文献需要用到很特殊的“//”，所以我又加了一个 after.slash。其他需要在特定符号后面输出，所以写了一个 output.after。

```
output.nonnull(s) ==
BEGIN
  s := argument on stack
  if output.state = mid.sentence then
    write$(pop() * ", ")
    -- "pop" isn't a function: just use stack top
  else
    if output.state = after.block then
      write$(add.period$(pop()))
      newline$
      write$("\newblock ")
    else
      if output.state = before.all then
        write$(pop())
      else
        -- output.state should be after.sentence
        write$(add.period$(pop()) * " ")
      fi
    fi
    output.state := mid.sentence
  fi
  push s on stack
END
```

The output function calls output.nonnull if its argument is non-empty; its argu-

ment may be a missing field (thus, not necessarily a string)

```
output(s) ==  
BEGIN  
    if not empty$(s) then output.nonnull(s)  
    fi  
END
```

The `output.check` function is the same as the `output` function except that, if necessary, `output.check` warns the user that the `t` field shouldn't be empty (this is because it probably won't be a good reference without the field; the entry functions try to make the formatting look reasonable even when such fields are empty).

```
output.check(s,t) ==  
BEGIN  
    if empty$(s) then  
        warning$("empty " * t * " in " * cite$)  
    else output.nonnull(s)  
    fi  
END
```

The `output.bibitem` function writes the `\bibitem` for the current entry (the label should already have been set up), and sets up the separator state for the output functions. And, it leaves a string on the stack as per the output convention.

```
output.bibitem ==  
BEGIN  
    newline$  
    write$("\bibitem[")      % for alphabetic labels,  
    write$(label)           % these three lines  
    write$("{")             % are used  
    write$("\bibitem{")      % this line for numeric labels  
    write$(cite$)  
    write$("}")  
    push "" on stack  
    output.state := before.all  
END
```

The `fin.entry` function finishes off an entry by adding a period to the string remaining on the stack. If the state is still `before.all` then nothing was produced for this entry, so the result will look bad, but the user deserves it. (We don't omit the whole entry because the entry was cited, and a `bibitem` is needed to define the citation label.)

```
fin.entry ==  
BEGIN  
    write$(add.period$(pop()))  
    newline$  
END
```

The `new.block` function prepares for a new block to be output, and `new.sentence` prepares for a new sentence.


```

new.block ==
BEGIN
    if output.state <> before.all then
        output.state := after.block
    fi
END

```

```

new.sentence ==
BEGIN
    if output.state <> after.block then
        if output.state <> before.all then
            output.state := after.sentence
        fi
    fi
END

```

```

573 INTEGERS { output.state before.all mid.sentence after.sentence after.block after.slash }
574
575 INTEGERS { lang.zh lang.ja lang.en lang.ru lang.other }
576
577 INTEGERS { charptr len }
578
579 FUNCTION {init.state.consts}
580 { #0 'before.all :=
581   #1 'mid.sentence :=
582   #2 'after.sentence :=
583   #3 'after.block :=
584   #4 'after.slash :=
585   #3 'lang.zh :=
586   #4 'lang.ja :=
587   #1 'lang.en :=
588   #2 'lang.ru :=
589   #0 'lang.other :=
590 }
591

```

下面是一些常量的定义

```

592 FUNCTION {bbl.anonymous}
593 { entry.lang lang.zh =
594   { "佚名" }
595   { "Anon" }
596   if$
597 }
598
599 FUNCTION {bbl.space}
600 { entry.lang lang.zh =
601   { "\ " }
602   { " " }
603   if$
604 }
605
606 FUNCTION {bbl.and}
607 { "" }
608

```

```

609 FUNCTION {bbl.et.al}
610 { entry.lang lang.zh =
611   { " 等" }
612   { entry.lang lang.ja =
613     { " 他" }
614     { entry.lang lang.ru =
615       { "идр" }
616       { "et~al." }
617       if$
618     }
619     if$
620   }
621   if$
622 }
623
624 FUNCTION {citation.and}
625 { terms.in.macro
626   { "{\biband}" }
627   'bbl.and
628   if$
629 }
630
631 FUNCTION {citation.et.al}
632 { terms.in.macro
633   { "{\bibetal}" }
634   'bbl.et.al
635   if$
636 }
637
638 FUNCTION {bbl.colon} { ": " }
639
640 FUNCTION {bbl.pages.colon}
641 { space.before.pages
642   { ": " }
643   { ":\allowbreak " }
644   if$
645 }
646
647 <*/2005>
648 FUNCTION {bbl.wide.space} { "\quad " }
649 </!2005>
650 <*/2005>
651 FUNCTION {bbl.wide.space} { "\ " }
652 </2005>
653
654 FUNCTION {bbl.slash} { "//\allowbreak " }
655
656 FUNCTION {bbl.sine.loco}
657 { entry.lang lang.zh =
658   { "[出版地不详]" }
659   { "[S.l.]" }
660   if$
661 }
662
663 FUNCTION {bbl.sine.nomine}

```

```

664 { entry.lang lang.zh =
665     { "[出版者不详]" }
666     { "[s.n.]" }
667     if$
668 }
669
670 FUNCTION {bbl.sine.loco.sine.nomine}
671 { entry.lang lang.zh =
672     { "[出版地不详: 出版者不详]" }
673     { "[S.l.: s.n.]" }
674     if$
675 }
676

```

These three functions pop one or two (integer) arguments from the stack and push a single one, either 0 or 1. The 'skip\$ in the 'and' and 'or' functions are used because the corresponding if\$ would be idempotent

```

677 FUNCTION {not}
678 { { #0 }
679   { #1 }
680   if$
681 }
682
683 FUNCTION {and}
684 { 'skip$
685   { pop$ #0 }
686   if$
687 }
688
689 FUNCTION {or}
690 { { pop$ #1 }
691   'skip$
692   if$
693 }
694
695 STRINGS { x y }
696
697 FUNCTION {contains}
698 { 'y :=
699   'x :=
700   y text.length$ 'len :=
701   x text.length$ len - #1 + 'charptr :=
702   { charptr #0 >
703     x charptr len substring$ y = not
704     and
705   }
706   { charptr #1 - 'charptr := }
707   while$
708   charptr #0 >
709 }
710

```

the variables s and t are temporary string holders

```

711 STRINGS { s t }

```

```

712
713 FUNCTION {output.nonnull}
714 { 's :=
715   output.state mid.sentence =
716   { ", " * write$ }
717   { output.state after.block =
718     { add.period$ write$
719       newline$
720       "\newblock " write$
721     }
722     { output.state before.all =
723       'write$
724       { output.state after.slash =
725         { bbl.slash * write$
726           newline$
727         }
728         { add.period$ " " * write$ }
729       }
730     }
731     if$
732   }
733   if$
734   mid.sentence 'output.state :=
735 }
736 if$
737 s
738 }
739
740 FUNCTION {output}
741 { duplicate$ empty$
742   'pop$
743   'output.nonnull
744   if$
745 }
746
747 FUNCTION {output.after}
748 { 't :=
749   duplicate$ empty$
750   'pop$
751   { 's :=
752     output.state mid.sentence =
753     { t * write$ }
754     { output.state after.block =
755       { add.period$ write$
756         newline$
757         "\newblock " write$
758       }
759       { output.state before.all =
760         'write$
761         { output.state after.slash =
762           { bbl.slash * write$ }
763           { add.period$ " " * write$ }
764         }
765         if$
766       }
767     }
768     if$

```

```

767         }
768         if$
769         mid.sentence 'output.state :=
770     }
771     if$
772     s
773 }
774 if$
775 }
776
777 FUNCTION {output.check}
778 { 't :=
779   duplicate$ empty$
780   { pop$ "empty " t * " in " * cite$ * warning$ }
781   'output.nonnull
782   if$
783 }
784

```

This function finishes all entries.

```

785 FUNCTION {fin.entry}
786 { end.with.period
787   'add.period$
788   'skip$
789   if$
790   write$
791   show.english.translation entry.lang lang.zh = and
792   { ""
793     write$
794   }
795   'skip$
796   if$
797   newline$
798 }
799
800 FUNCTION {new.block}
801 { output.state before.all =
802   'skip$
803   { output.state after.slash =
804     'skip$
805     { after.block 'output.state := }
806     if$
807   }
808   if$
809 }
810
811 FUNCTION {new.sentence}
812 { output.state after.block =
813   'skip$
814   { output.state before.all =
815     'skip$
816     { output.state after.slash =
817       'skip$
818       { after.sentence 'output.state := }
819       if$

```

```

820     }
821     if$
822   }
823   if$
824 }
825
826 FUNCTION {new.slash}
827 { output.state before.all =
828   'skip$
829   { slash.for.extraction
830     { after.slash 'output.state := }
831     { after.block 'output.state := }
832     if$
833   }
834   if$
835 }
836

```

Sometimes we begin a new block only if the block will be big enough. The new.block.checka function issues a new.block if its argument is nonempty; new.block.checkb does the same if either of its TWO arguments is nonempty.

```

837 FUNCTION {new.block.checka}
838 { empty$
839   'skip$
840   'new.block
841   if$
842 }
843
844 FUNCTION {new.block.checkb}
845 { empty$
846   swap$ empty$
847   and
848   'skip$
849   'new.block
850   if$
851 }
852

```

The new.sentence.check functions are analogous.

```

853 FUNCTION {new.sentence.checka}
854 { empty$
855   'skip$
856   'new.sentence
857   if$
858 }
859
860 FUNCTION {new.sentence.checkb}
861 { empty$
862   swap$ empty$
863   and
864   'skip$
865   'new.sentence
866   if$
867 }

```

B.4 Formatting chunks

Here are some functions for formatting chunks of an entry. By convention they either produce a string that can be followed by a comma or period (using `add.period$`, so it is OK to end in a period), or they produce the null string.

A useful utility is the `field.or.null` function, which checks if the argument is the result of pushing a ‘missing’ field (one for which no assignment was made when the current entry was read in from the database) or the result of pushing a string having no non-white-space characters. It returns the null string if so, otherwise it returns the field string. Its main (but not only) purpose is to guarantee that what’s left on the stack is a string rather than a missing field.

```
field.or.null(s) ==
BEGIN
    if empty$(s) then return ""
    else return s
END
```

Another helper function is `emphasize`, which returns the argument emphasised, if that is non-empty, otherwise it returns the null string. Italic corrections aren’t used, so this function should be used when punctuation will follow the result.

```
emphasize(s) ==
BEGIN
    if empty$(s) then return ""
    else return "{\em " * s * "}"
```

The ‘`pop$`’ in this function gets rid of the duplicate ‘empty’ value and the ‘`skip$`’ returns the duplicate field value

```
869 FUNCTION {field.or.null}
870 { duplicate$ empty$
871   { pop$ "" }
872   'skip$
873   if$
874 }
875
876 FUNCTION {emphasize}
877 { duplicate$ empty$
878   { pop$ "" }
879   { "\emph{" swap$ * "}" * }
880   if$
881 }
882
883 FUNCTION {format.btitle}
884 { italic.book.title
885   entry.lang lang.en = and
```

```

886     'emphasize
887     'skip$
888   if$
889 }
890

```

B.4.1 Detect Language

```

891 INTEGERS { byte second.byte }
892
893 INTEGERS { char.lang tmp.lang }
894
895 STRINGS { tmp.str }
896
897 FUNCTION {get.str.lang}
898 { 'tmp.str :=
899   lang.other 'tmp.lang :=
900   #1 'charptr :=
901   tmp.str text.length$ #1 + 'len :=
902   { charptr len < }
903   { tmp.str charptr #1 substring$ chr.to.int$ 'byte :=
904     byte #128 <
905     { charptr #1 + 'charptr :=
906       byte #64 > byte #91 < and byte #96 > byte #123 < and or
907       { lang.en 'char.lang := }
908       { lang.other 'char.lang := }
909     if$
910   }
911   { tmp.str charptr #1 + #1 substring$ chr.to.int$ 'second.byte :=
912     byte #224 <

```

俄文西里尔字母: U+0400 到 U+052F, 对应 UTF-8 从 D0 80 到 D4 AF.

```

913     { charptr #2 + 'charptr :=
914       byte #207 > byte #212 < and
915       byte #212 = second.byte #176 < and or
916       { lang.ru 'char.lang := }
917       { lang.other 'char.lang := }
918     if$
919   }
920   { byte #240 <

```

CJK Unified Ideographs: U+4E00–U+9FFF; UTF-8: E4 B8 80–E9 BF BF.

```

921     { charptr #3 + 'charptr :=
922       byte #227 > byte #234 < and
923       { lang.zh 'char.lang := }

```

CJK Unified Ideographs Extension A: U+3400–U+4DBF; UTF-8: E3 90 80–E4 B6 BF.

```

924     { byte #227 =
925       { second.byte #143 >
926         { lang.zh 'char.lang := }

```

日语假名: U+3040–U+30FF, UTF-8: E3 81 80–E3 83 BF.

```

927     { second.byte #128 > second.byte #132 < and
928     { lang.ja 'char.lang := }

```



```

929             { lang.other 'char.lang := }
930             if$
931         }
932     if$
933 }

```

CJK Compatibility Ideographs: U+F900–U+FAFF, UTF-8: EF A4 80–EF AB BF.

```

934     { byte #239 =
935         second.byte #163 > second.byte #172 < and and
936         { lang.zh 'char.lang := }
937         { lang.other 'char.lang := }
938     if$
939 }
940 if$
941 }
942 if$
943 }

```

CJK Unified Ideographs Extension B–F: U+20000–U+2EBEF, UTF-8: F0 A0 80 80–F0 AE AF AF. CJK Compatibility Ideographs Supplement: U+2F800–U+2FA1F, UTF-8: F0 AF A0 80–F0 AF A8 9F.

```

944     { charptr #4 + 'charptr :=
945     byte #240 = second.byte #159 > and
946     { lang.zh 'char.lang := }
947     { lang.other 'char.lang := }
948     if$
949 }
950 if$
951 }
952 if$
953 }
954 if$
955 char.lang tmp.lang >
956 { char.lang 'tmp.lang := }
957 'skip$
958 if$
959 }
960 while$
961 tmp.lang
962 }
963
964 FUNCTION {check.entry.lang}
965 { author field.or.null
966   title field.or.null *
967   get.str.lang
968 }
969
970 STRINGS { entry.langid }
971
972 FUNCTION {set.entry.lang}
973 { "" 'entry.langid :=
974   language empty$ not
975   { language 'entry.langid := }
976   'skip$

```

```

977 if$
978   langid empty$ not
979   { langid 'entry.langid := }
980   'skip$
981 if$
982 entry.langid empty$
983   { check.entry.lang }
984   { entry.langid "english" = entry.langid "american" = or entry.langid "british" = or
985     { lang.en }
986     { entry.langid "chinese" =
987       { lang.zh }
988       { entry.langid "japanese" =
989         { lang.ja }
990         { entry.langid "russian" =
991           { lang.ru }
992           { check.entry.lang }
993         if$
994       }
995     }
996   }
997   if$
998   }
999   if$
1000  }
1001 if$
1002 'entry.lang :=
1003 }
1004
1005 FUNCTION {set.entry.numbered}
1006 { type$ "patent" =
1007   type$ "standard" = or
1008   type$ "techreport" = or
1009   { #1 'entry.numbered := }
1010   { #0 'entry.numbered := }
1011   if$
1012 }
1013

```

B.4.2 Format names

The `format.names` function formats the argument (which should be in BibTeX name format) into "First Von Last, Junior", separated by commas and with an "and" before the last (but ending with "et al." if the last of multiple authors is "others"). This function's argument should always contain at least one name.

```

VAR: nameptr, namesleft, numnames: INTEGER
pseudoVAR: nameresult: STRING      (it's what's accumulated on the stack)

format.names(s) ==
BEGIN
  nameptr := 1
  numnames := num.names$(s)
  namesleft := numnames
  while namesleft > 0

```

```

do
    % for full names:
    t := format.name$(s, nameptr, "{ff~}{vv~}{ll}{, jj}")
    % for abbreviated first names:
    t := format.name$(s, nameptr, "{f.~}{vv~}{ll}{, jj}")
    if nameptr > 1 then
        if namesleft > 1 then namerresult := namerresult * ", " * t
        else if numnames > 2
            then namerresult := namerresult * ","
            fi
        if t = "others"
            then namerresult := namerresult * " et~al."
            else namerresult := namerresult * " and " * t
            fi
        fi
    else namerresult := t
    fi
    nameptr := nameptr + 1
    namesleft := namesleft - 1
od
return namerresult
END

```

The `format.authors` function returns the result of `format.names(author)` if the author is present, or else it returns the null string

```

format.authors ==
BEGIN
    if empty$(author) then return ""
    else return format.names(author)
    fi
END

```

`Format.editors` is like `format.authors`, but it uses the editor field, and appends ", editor" or ", editors"

```

format.editors ==
BEGIN
    if empty$(editor) then return ""
    else
        if num.names$(editor) > 1 then
            return format.names(editor) * ", editors"
        else
            return format.names(editor) * ", editor"
        fi
    fi
END

```

Other formatting functions are similar, so no "comment version" will be given for them.

```

1014 INTEGERS { nameptr namesleft numnames name.lang }
1015
1016 FUNCTION {format.name}

```

```

1017 { "{vv~}{ll}{, jj}{, ff}" format.names$ 't :=
1018   t "others" =
1019     { bbl.et.al }
1020     { t get.str.lang 'name.lang :=
1021       name.lang lang.en =
1022         { t #1 "{vv~}{ll}{ f{~}}" format.names$
1023           uppercase.name
1024             { "u" change.case$ }
1025             'skip$
1026             if$
1027               t #1 "{, jj}" format.names$ *
1028             }
1029           { t #1 "{ll}{ff}" format.names$ }
1030         if$
1031       }
1032   if$
1033 }
1034
1035 FUNCTION {format.names}
1036 { 's :=
1037   #1 'nameptr :=
1038   s num.names$ 'numnames :=
1039   ""
1040   numnames 'namesleft :=
1041     { namesleft #0 > }
1042     { s nameptr format.name bbl.et.al =
1043       numnames bibliography.et.al.min #1 - > nameptr bibliography.et.al.use.first > and or
1044       { ", " *
1045         bbl.et.al *
1046         #1 'namesleft :=
1047       }
1048       { nameptr #1 >
1049         { namesleft #1 = bbl.and "" = not and
1050           { bbl.and * }
1051           { ", " * }
1052         if$
1053       }
1054       'skip$
1055       if$
1056       s nameptr format.name *
1057     }
1058   if$
1059   nameptr #1 + 'nameptr :=
1060   namesleft #1 - 'namesleft :=
1061   }
1062   while$
1063 }
1064
1065 FUNCTION {format.key}
1066 { empty$
1067   { key field.or.null }
1068   { "" }
1069   if$
1070 }
1071

```

```

1072 FUNCTION {format.authors}
1073 { author empty$ not
1074   { author format.names }
1075   { "empty author in " cite$ * warning$
1076 <*author-year>
1077   bbl.anonymous
1078 </author-year>
1079 <*numerical>
1080   ""
1081 </numerical>
1082   }
1083   if$
1084 }
1085
1086 FUNCTION {format.editors}
1087 { editor empty$
1088   { "" }
1089   { editor format.names }
1090   if$
1091 }
1092
1093 FUNCTION {format.translators}
1094 { translator empty$
1095   { "" }
1096   { translator format.names
1097     entry.lang lang.zh =
1098     { translator num.names$ #3 >
1099       { " 译" * }
1100       { ", 译" * }
1101       if$
1102     }
1103     'skip$
1104     if$
1105   }
1106   if$
1107 }
1108
1109 FUNCTION {format.full.names}
1110 {'s :=
1111 #1 'nameptr :=
1112 s num.names$ 'numnames :=
1113 numnames 'namesleft :=
1114 { namesleft #0 > }
1115 { s nameptr "{vv~}{ll}{, jj}{, ff}" format.name$ 't :=
1116   t get.str.lang 'name.lang :=
1117   name.lang lang.en =
1118   { t #1 "{vv~}{ll}" format.name$ 't := }
1119   { t #1 "{ll}{ff}" format.name$ 't := }
1120   if$
1121   nameptr #1 >
1122   {
1123     namesleft #1 >
1124     { ", " * t * }
1125     {
1126       numnames #2 >

```

```

1127         { "," * }
1128         'skip$
1129         if$
1130         t "others" =
1131         { " et~al." * }
1132         { " and " * t * }
1133         if$
1134     }
1135     if$
1136 }
1137 't
1138 if$
1139     nameptr #1 + 'nameptr :=
1140     namesleft #1 - 'namesleft :=
1141 }
1142 while$
1143 }
1144
1145 FUNCTION {author.editor.full}
1146 { author empty$
1147   { editor empty$
1148     { "" }
1149     { editor format.full.names }
1150   if$
1151   }
1152   { author format.full.names }
1153 if$
1154 }
1155
1156 FUNCTION {author.full}
1157 { author empty$
1158   { "" }
1159   { author format.full.names }
1160 if$
1161 }
1162
1163 FUNCTION {editor.full}
1164 { editor empty$
1165   { "" }
1166   { editor format.full.names }
1167 if$
1168 }
1169
1170 FUNCTION {make.full.names}
1171 { type$ "book" =
1172   type$ "inbook" =
1173   or
1174   'author.editor.full
1175   { type$ "collection" =
1176     type$ "proceedings" =
1177     or
1178     'editor.full
1179     'author.full
1180   if$
1181 }

```

```

1182 if$
1183 }
1184
1185 FUNCTION {output.bibitem}
1186 { newline$
1187   "\bibitem[" write$
1188   label "]" *
1189   make.full.names duplicate$ short.list =
1190   { pop$ }
1191   { duplicate$ "[" contains
1192     { "{" swap$ * "]" * }
1193     'skip$
1194     if$
1195     *
1196   }
1197   if$
1198   "]"{" * write$
1199   cite$ write$
1200   "]" write$
1201   newline$
1202   ""
1203   before.all 'output.state :=
1204 }
1205

```

B.4.3 Format title

The `format.title` function is used for non-book-like titles. For most styles we convert to lowercase (except for the very first letter, and except for the first one after a colon (followed by whitespace)), and hope the user has brace-surrounded words that need to stay capitilized; for some styles, however, we leave it as it is in the database.

```

1206 FUNCTION {change.sentence.case}
1207 { entry.lang lang.en =
1208   { "t" change.case$ }
1209   'skip$
1210   if$
1211 }
1212
1213 FUNCTION {add.link}
1214 { url empty$ not
1215   { "\href{" url * "{" * swap$ * "]" * }
1216   { doi empty$ not
1217     { "\href{https://doi.org/" doi * "{" * swap$ * "]" * }
1218     'skip$
1219     if$
1220   }
1221   if$
1222 }
1223
1224 FUNCTION {format.title}
1225 { title empty$
1226   { "" }

```

```

1227 { title
1228     sentence.case.title
1229     'change.sentence.case
1230     'skip$
1231     if$
1232     entry.numbered number empty$ not and
1233     { bbl.colon *
1234         type$ "patent" = show.patent.country and
1235         { address empty$ not
1236             { address * ", " * }
1237             { location empty$ not
1238                 { location * ", " * }
1239                 { entry.lang lang.zh =
1240                     { " 中国" * ", " * }
1241                     'skip$
1242                     if$
1243                     }
1244                     if$
1245                     }
1246                     if$
1247                     }
1248                     'skip$
1249                     if$
1250                     number *
1251                     }
1252                     'skip$
1253                     if$
1254                     link.title
1255                     'add.link
1256                     'skip$
1257                     if$
1258                     }
1259     if$
1260 }
1261

```

For several functions we'll need to connect two strings with a tie (~) if the second one isn't very long (fewer than 3 characters). The tie.or.space.connect function does that. It concatenates the two strings on top of the stack, along with either a tie or space between them, and puts this concatenation back onto the stack:

```

tie.or.space.connect(str1,str2) ==
BEGIN
    if text.length$(str2) < 3
    then return the concatenation of str1, "~", and str2
    else return the concatenation of str1, " ", and str2
END

```

```

1262 FUNCTION {tie.or.space.connect}
1263 { duplicate$ text.length$ #3 <
1264   { "~" }
1265   { " " }
1266   if$
1267   swap$ * *

```



```
1268 }
1269
```

The `either.or.check` function complains if both fields or an either-or pair are nonempty.

```
either.or.check(t,s) ==
BEGIN
  if empty$(s) then
    warning$(can't use both " * t * " fields in " * cite$)
  fi
END
```

```
1270 FUNCTION {either.or.check}
1271 { empty$
1272   'pop$
1273   { "can't use both " swap$ * " fields in " * cite$ * warning$ }
1274   if$
1275 }
1276
```

The `format.bvolume` function is for formatting the volume and perhaps series name of a multivolume work. If both a volume and a series field are there, we assume the series field is the title of the whole multivolume work (the title field should be the title of the thing being referred to), and we add an "of <series>". This function is called in mid-sentence.

The `format.number.series` function is for formatting the series name and perhaps number of a work in a series. This function is similar to `format.bvolume`, although for this one the series must exist (and the volume must not exist). If the number field is empty we output either the series field unchanged if it exists or else the null string. If both the number and series fields are there we assume the series field gives the name of the whole series (the title field should be the title of the work being one referred to), and we add an "in <series>". We capitalize Number when this function is used at the beginning of a block.

```
1277 FUNCTION {is.digit}
1278 { duplicate$ empty$
1279   { pop$ #0 }
1280   { chr.to.int$
1281     duplicate$ "0" chr.to.int$ <
1282     { pop$ #0 }
1283     { "9" chr.to.int$ >
1284       { #0 }
1285       { #1 }
1286     if$
1287   }
1288   if$
1289 }
1290 if$
1291 }
```

```

1292
1293 FUNCTION {is.number}
1294 { 's :=
1295   s empty$
1296   { #0 }
1297   { s text.length$ 'charptr :=
1298     { charptr #0 >
1299       s charptr #1 substring$ is.digit
1300       and
1301     }
1302     { charptr #1 - 'charptr := }
1303     while$
1304     charptr not
1305   }
1306   if$
1307 }
1308
1309 FUNCTION {format.volume}
1310 { volume empty$ not
1311   { volume is.number
1312     { entry.lang lang.zh =
1313       { " 第 " volume * " 卷" * }
1314       { "volume" volume tie.or.space.connect }
1315     if$
1316   }
1317   { volume }
1318   if$
1319 }
1320 { "" }
1321 if$
1322 }
1323
1324 FUNCTION {format.number}
1325 { number empty$ not
1326   { number is.number
1327     { entry.lang lang.zh =
1328       { " 第 " number * " 册" * }
1329       { "number" number tie.or.space.connect }
1330     if$
1331   }
1332   { number }
1333   if$
1334 }
1335 { "" }
1336 if$
1337 }
1338
1339 FUNCTION {format.volume.number}
1340 { volume empty$ not
1341   { format.volume }
1342   { format.number }
1343   if$
1344 }
1345
1346 FUNCTION {format.title.vol.num}

```

```

1347 { title
1348   sentence.case.title
1349   'change.sentence.case
1350   'skip$
1351   if$
1352   entry.numbered
1353     { number empty$ not
1354       { bbl.colon * number * }
1355       'skip$
1356       if$
1357     }
1358     { format.volume.number 's :=
1359       s empty$ not
1360       { bbl.colon * s * }
1361       'skip$
1362       if$
1363     }
1364   if$
1365 }
1366
1367 FUNCTION {format.series.vol.num.title}
1368 { format.volume.number 's :=
1369   series empty$ not
1370   { series
1371     sentence.case.title
1372     'change.sentence.case
1373     'skip$
1374     if$
1375     entry.numbered
1376       { bbl.wide.space * }
1377       { bbl.colon *
1378         s empty$ not
1379         { s * bbl.wide.space * }
1380         'skip$
1381         if$
1382       }
1383     if$
1384     title *
1385     sentence.case.title
1386     'change.sentence.case
1387     'skip$
1388     if$
1389     entry.numbered number empty$ not and
1390       { bbl.colon * number * }
1391       'skip$
1392     if$
1393   }
1394   { format.title.vol.num }
1395   if$
1396   format.btitle
1397   link.title
1398   'add.link
1399   'skip$
1400   if$
1401 }

```

```

1402
1403 FUNCTION {format.booktitle.vol.num}
1404 { booktitle
1405   entry.numbered
1406   'skip$
1407   { format.volume.number 's :=
1408     s empty$ not
1409     { bbl.colon * s * }
1410     'skip$
1411     if$
1412   }
1413   if$
1414 }
1415
1416 FUNCTION {format.series.vol.num.booktitle}
1417 { format.volume.number 's :=
1418   series empty$ not
1419   { series bbl.colon *
1420     entry.numbered not s empty$ not and
1421     { s * bbl.wide.space * }
1422     'skip$
1423     if$
1424     booktitle *
1425   }
1426   { format.booktitle.vol.num }
1427   if$
1428   format.btitle
1429   in.booktitle
1430   { duplicate$ empty$ not entry.lang lang.en = and
1431     { "In: " swap$ * }
1432     'skip$
1433     if$
1434   }
1435   'skip$
1436   if$
1437 }
1438
1439 FUNCTION {remove.period}
1440 { 't :=
1441   "" 's :=
1442   { t empty$ not }
1443   { t #1 #1 substring$ 'tmp.str :=
1444     tmp.str "." = not
1445     { s tmp.str * 's := }
1446     'skip$
1447     if$
1448     t #2 global.max$ substring$ 't :=
1449   }
1450   while$
1451   s
1452 }
1453
1454 FUNCTION {abbreviate}
1455 { remove.period
1456   't :=

```

```

1457 t "l" change.case$ 's :=
1458 ""
1459 s "physical review letters" =
1460 { "Phys Rev Lett" }
1461 'skip$
1462 if$
1463 's :=
1464 s empty$
1465 { t }
1466 { pop$ s }
1467 if$
1468 }
1469
1470 FUNCTION {get.journal.title}
1471 { short.journal
1472   { shortjournal empty$ not
1473     { shortjournal }
1474     { journal empty$ not
1475       { journal abbreviate }
1476       { journaltitle empty$ not
1477         { journaltitle abbreviate }
1478         { "" }
1479         if$
1480       }
1481       if$
1482     }
1483     if$
1484   }
1485   { journal empty$ not
1486     { journal }
1487     { journaltitle empty$ not
1488       { journaltitle }
1489       { shortjournal empty$ not
1490         { shortjournal }
1491         { "" }
1492         if$
1493       }
1494       if$
1495     }
1496     if$
1497   }
1498   if$
1499 }
1500
1501 FUNCTION {check.arxiv.preprint}
1502 { #1 #5 substring$ "l" change.case$ "arxiv" =
1503   { #1 }
1504   { #0 }
1505   if$
1506 }
1507
1508 FUNCTION {format.journal}
1509 { get.journal.title
1510   duplicate$ empty$ not
1511   { italic.journal entry.lang lang.en = and

```

```

1512         'emphasize
1513         'skip$
1514     if$
1515 }
1516 'skip$
1517 if$
1518 }
1519

```

B.4.4 Format entry type mark

```

1520 FUNCTION {set.entry.mark}
1521 { entry.mark empty$ not
1522   'pop$
1523   { mark empty$ not
1524     { pop$ mark 'entry.mark := }
1525     { 'entry.mark := }
1526     if$
1527   }
1528   if$
1529 }
1530
1531 FUNCTION {format.mark}
1532 { show.mark
1533   { entry.mark
1534     show.medium.type
1535     { medium empty$ not
1536       { "/" * medium * }
1537       { entry.is.electronic
1538         { "/OL" * }
1539         'skip$
1540         if$
1541       }
1542       if$
1543     }
1544     'skip$
1545     if$
1546     'entry.mark :=
1547     space.before.mark
1548     { " " }
1549     { "\allowbreak" }
1550     if$
1551     "[" * entry.mark * "]" *
1552   }
1553   { "" }
1554   if$
1555 }
1556

```

B.4.5 Format edition

The `format.edition` function appends " edition" to the edition, if present. We lowercase the edition (it should be something like "Third"), because this doesn't start a sentence.

```

1557 FUNCTION {num.to.ordinal}
1558 { duplicate$ text.length$ 'charptr :=
1559   duplicate$ charptr #1 substring$ 's :=
1560   s "1" =
1561     { "st" * }
1562   { s "2" =
1563     { "nd" * }
1564     { s "3" =
1565       { "rd" * }
1566       { "th" * }
1567       if$
1568     }
1569     if$
1570   }
1571   if$
1572 }
1573
1574 FUNCTION {format.edition}
1575 { edition empty$
1576   { "" }
1577   { edition is.number
1578     { entry.lang lang.zh =
1579       { edition " 版" * }
1580       { edition num.to.ordinal " ed." * }
1581       if$
1582     }
1583     { entry.lang lang.en =
1584       { edition change.sentence.case 's :=
1585         s "Revised" = s "Revised edition" = or
1586         { "Rev. ed." }
1587         { s " ed." * }
1588         if$
1589       }
1590       { edition }
1591       if$
1592     }
1593     if$
1594   }
1595   if$
1596 }
1597

```

B.4.6 Format publishing items

出版地址和出版社会有“[S.l.: s.n.]”的情况，所以必须一起处理。

```

1598 FUNCTION {format.publisher}
1599 { publisher empty$ not
1600   { publisher }
1601   { school empty$ not
1602     { school }
1603     { organization empty$ not
1604       { organization }
1605       { institution empty$ not
1606         { institution }

```

```

1607         { "" }
1608         if$
1609     }
1610     if$
1611 }
1612 if$
1613 }
1614 if$
1615 }
1616
1617 FUNCTION {format.address.publisher}
1618 { address empty$ not
1619   { address }
1620   { location empty$ not
1621     { location }
1622     { "" }
1623     if$
1624   }
1625   if$
1626   duplicates$ empty$ not
1627   { format.publisher empty$ not
1628     { bbl.colon * format.publisher * }
1629     { entry.is.electronic not show.missing.address.publisher and
1630       { bbl.colon * bbl.sine.nomine * }
1631       'skip$
1632       if$
1633     }
1634     if$
1635   }
1636   { pop$
1637     entry.is.electronic not show.missing.address.publisher and
1638     { format.publisher empty$ not
1639       { bbl.sine.loco bbl.colon * format.publisher * }
1640       { bbl.sine.loco.sine.nomine }
1641       if$
1642     }
1643     { format.publisher empty$ not
1644       { format.publisher }
1645       { "" }
1646       if$
1647     }
1648     if$
1649   }
1650   if$
1651 }
1652

```

B.4.7 Format date

The `format.date` function is for the month and year, but we give a warning if there's an empty year but the month is there, and we return the empty string if they're both empty.

期刊需要著录起止范围，其中年份使用“/”分隔，卷和期使用“-”分隔。版

本 v2.0.2 前的年份也使用“-”分隔，仅提供兼容性，不再推荐。

```
1653 FUNCTION {extract.before.dash}
1654 { duplicate$ empty$
1655   { pop$ "" }
1656   { 's :=
1657     #1 'charptr :=
1658     s text.length$ #1 + 'len :=
1659     { charptr len <
1660       s charptr #1 substring$ "-" = not
1661       and
1662     }
1663     { charptr #1 + 'charptr := }
1664     while$
1665     s #1 charptr #1 - substring$
1666   }
1667   if$
1668 }
1669
1670 FUNCTION {extract.after.dash}
1671 { duplicate$ empty$
1672   { pop$ "" }
1673   { 's :=
1674     #1 'charptr :=
1675     s text.length$ #1 + 'len :=
1676     { charptr len <
1677       s charptr #1 substring$ "-" = not
1678       and
1679     }
1680     { charptr #1 + 'charptr := }
1681     while$
1682     { charptr len <
1683       s charptr #1 substring$ "-" =
1684       and
1685     }
1686     { charptr #1 + 'charptr := }
1687     while$
1688     s charptr global.max$ substring$
1689   }
1690   if$
1691 }
1692
1693 FUNCTION {extract.before.slash}
1694 { duplicate$ empty$
1695   { pop$ "" }
1696   { 's :=
1697     #1 'charptr :=
1698     s text.length$ #1 + 'len :=
1699     { charptr len <
1700       s charptr #1 substring$ "/" = not
1701       and
1702     }
1703     { charptr #1 + 'charptr := }
1704     while$
1705     s #1 charptr #1 - substring$
1706   }
```

```

1707   if$
1708 }
1709
1710 FUNCTION {extract.after.slash}
1711 { duplicate$ empty$
1712   { pop$ "" }
1713   { 's :=
1714     #1 'charptr :=
1715     s text.length$ #1 + 'len :=
1716     { charptr len <
1717       s charptr #1 substring$ "-" = not
1718       and
1719       s charptr #1 substring$ "/" = not
1720       and
1721     }
1722     { charptr #1 + 'charptr := }
1723     while$
1724     { charptr len <
1725       s charptr #1 substring$ "-" =
1726       s charptr #1 substring$ "/" =
1727       or
1728       and
1729     }
1730     { charptr #1 + 'charptr := }
1731     while$
1732     s charptr global.max$ substring$
1733   }
1734   if$
1735 }
1736

```

著者-出版年制必须提取出年份

```

1737 FUNCTION {format.year}
1738 { year empty$ not
1739   { year extract.before.slash extra.label * }
1740   { date empty$ not
1741     { date extract.before.dash extra.label * }
1742     { "empty year in " cite$ * warning$
1743       urldate empty$ not
1744       { "[" urldate extract.before.dash * extra.label * "]" * }
1745       { "" }
1746     }
1747     if$
1748   }
1749   if$
1750 }
1751 }
1752
1753 FUNCTION {format.periodical.year}
1754 { year empty$ not
1755   { year extract.before.slash
1756     "--" *
1757     year extract.after.slash
1758     duplicate$ empty$
1759     'pop$

```

```

1760         { * }
1761     if$
1762 }
1763 { date empty$ not
1764     { date extract.before.dash }
1765     { "empty year in " cite$ * warning$
1766         urldate empty$ not
1767         { "[" urldate extract.before.dash * "]" * }
1768         { "" }
1769     if$
1770 }
1771 if$
1772 }
1773 if$
1774 }
1775

```

专利和报纸都是使用日期而不是年

```

1776 FUNCTION {format.date}
1777 { date empty$ not
1778     { type$ "patent" = type$ "newspaper" = or
1779         { date }
1780         { format.year }
1781     if$
1782 }
1783 { year empty$ not
1784     { format.year }
1785     { "" }
1786 if$
1787 }
1788 if$
1789 }
1790

```

更新、修改日期只用于电子资源 electronic

```

1791 FUNCTION {format.editdate}
1792 { date empty$ not
1793     { "\allowbreak(" date * ")" * }
1794     { "" }
1795 if$
1796 }
1797

```

国标中的“引用日期”都是与 URL 同时出现的，所以其实为 urldate，这个虽然不是 Bib_TE_X 标准的域，但是实际中很常见。

```

1798 FUNCTION {format.urldate}
1799 { show.urldate show.url and entry.url empty$ not and
1800     is.pure.electronic or
1801     urldate empty$ not and
1802     { "\allowbreak(" urldate * ")" * }
1803     { "" }
1804 if$
1805 }
1806

```

B.4.8 Format pages

By default, BibTeX sets the global integer variable `global.max$` to the BibTeX constant `glob_str_size`, the maximum length of a global string variable. Analogously, BibTeX sets the global integer variable `entry.max$` to `ent_str_size`, the maximum length of an entry string variable. The style designer may change these if necessary (but this is unlikely)

The `n.dashify` function makes each single ``-'` in a string a double ``--'` if it's not already

```
pseudoVAR: pageresult: STRING      (it's what's accumulated on the stack)

n.dashify(s) ==
BEGIN
  t := s
  pageresult := ""
  while (not empty$(t))
  do
    if (first character of t = "-")
    then
      if (next character isn't)
      then
        pageresult := pageresult * "--"
        t := t with the "-" removed
      else
        while (first character of t = "-")
        do
          pageresult := pageresult * "--"
          t := t with the "-" removed
        od
      fi
    else
      pageresult := pageresult * the first character
      t := t with the first character removed
    fi
  od
  return pageresult
END
```

国标里页码范围的连接号使用 `hyphen`，需要将 `dash` 转为 `hyphen`。

```
1807 FUNCTION {hyphenate}
1808 { 't :=
1809   ""
1810   { t empty$ not }
1811   { t #1 #1 substring$ "-" =
1812     { wave.dash.in.pages
1813       { "~" * }
1814       { "-" * }
1815       if$
1816       { t #1 #1 substring$ "-" = }
1817       { t #2 global.max$ substring$ 't := }
```

```

1818         while$
1819     }
1820     { t #1 #1 substring$ *
1821       t #2 global.max$ substring$ 't :=
1822     }
1823     if$
1824 }
1825 while$
1826 }
1827

```

This function doesn't begin a sentence so "pages" isn't capitalized. Other functions that use this should keep that in mind.

```

1828 FUNCTION {format.pages}
1829 { pages empty$
1830   { "" }
1831   { pages hyphenate }
1832   if$
1833 }
1834
1835 FUNCTION {format.extracted.pages}
1836 { pages empty$
1837   { "" }
1838   { pages
1839     only.start.page
1840     'extract.before.dash
1841     'hyphenate
1842     if$
1843   }
1844   if$
1845 }
1846

```

The `format.vol.num.pages` function is for the volume, number, and page range of a journal article. We use the format: `vol(number):pages`, with some variations for empty fields. This doesn't begin a sentence.

报纸在卷号缺失时，期号与前面的日期直接相连，所以必须拆开输出。

```

1847 FUNCTION {format.journal.volume}
1848 { volume empty$ not
1849   { bold.journal.volume
1850     { "\textbf{" volume * "}" * }
1851     { volume }
1852     if$
1853   }
1854   { "" }
1855   if$
1856 }
1857
1858 FUNCTION {format.journal.number}
1859 { number empty$ not
1860   { "\allowbreak (" number * ")" * }
1861   { "" }
1862   if$

```

```

1863 }
1864
1865 FUNCTION {format.journal.pages}
1866 { pages empty$
1867   { "" }
1868   { format.extracted.pages }
1869   if$
1870 }
1871

```

连续出版物的年卷期有起止范围，需要特殊处理

```

1872 FUNCTION {format.periodical.year.volume.number}
1873 { year empty$ not
1874   { year extract.before.slash }
1875   { "empty year in periodical " cite$ * warning$ }
1876   if$
1877   volume empty$ not
1878   { ", " * volume extract.before.dash * }
1879   'skip$
1880   if$
1881   number empty$ not
1882   { "\allowbreak (" * number extract.before.dash * ")" * }
1883   'skip$
1884   if$
1885   "--" *
1886   year extract.after.slash empty$
1887   volume extract.after.dash empty$ and
1888   number extract.after.dash empty$ and not
1889   { year extract.after.slash empty$ not
1890     { year extract.after.slash * }
1891     { year extract.before.slash * }
1892     if$
1893     volume empty$ not
1894     { ", " * volume extract.after.dash * }
1895     'skip$
1896     if$
1897     number empty$ not
1898     { "\allowbreak (" * number extract.after.dash * ")" * }
1899     'skip$
1900     if$
1901   }
1902   'skip$
1903   if$
1904 }
1905

```

B.4.9 Format url and doi

传统的 Bib_T_EX 习惯使用 howpublished 著录 url，这里提供支持。

```

1906 FUNCTION {check.url}
1907 { url empty$ not
1908   { "\url{" url * "}" * 'entry.url :=
1909     #1 'entry.is.electronic :=
1910   }

```

```

1911 { howpublished empty$ not
1912   { howpublished #1 #5 substring$ "\url{" =
1913     { howpublished 'entry.url :=
1914       #1 'entry.is.electronic :=
1915     }
1916     'skip$
1917   if$
1918 }
1919 { note empty$ not
1920   { note #1 #5 substring$ "\url{" =
1921     { note 'entry.url :=
1922       #1 'entry.is.electronic :=
1923     }
1924     'skip$
1925   if$
1926 }
1927 'skip$
1928 if$
1929 }
1930 if$
1931 }
1932 if$
1933 }
1934
1935 FUNCTION {output.url}
1936 { show.url is.pure.electronic or
1937   entry.url empty$ not and
1938   { new.block
1939     entry.url output
1940   }
1941   'skip$
1942   if$
1943 }
1944

```

需要检测 DOI 是否已经包含在 URL 中。

```

1945 FUNCTION {check.doi}
1946 { doi empty$ not
1947   { #1 'entry.is.electronic := }
1948   'skip$
1949   if$
1950 }
1951
1952 FUNCTION {is.in.url}
1953 { 's :=
1954   s empty$
1955   { #1 }
1956   { entry.url empty$
1957     { #0 }
1958     { s text.length$ 'len :=
1959       entry.url text.length$ 'charptr :=
1960       { entry.url charptr len substring$ s = not
1961         charptr #0 >
1962         and
1963       }

```

```

1964         { charptr #1 - 'charptr := }
1965         while$
1966         charptr
1967     }
1968     if$
1969 }
1970 if$
1971 }
1972
1973 FUNCTION {format.doi}
1974 { ""
1975   doi empty$ not
1976   { "" 's :=
1977     doi 't :=
1978     #0 'numnames :=
1979     { t empty$ not}
1980     { t #1 #1 substring$ 'tmp.str :=
1981       tmp.str "," = tmp.str " " = or t #2 #1 substring$ empty$ or
1982       { t #2 #1 substring$ empty$
1983         { s tmp.str * 's := }
1984         'skip$
1985         if$
1986         s empty$ s is.in.url or
1987         'skip$
1988         { numnames #1 + 'numnames :=
1989           numnames #1 >
1990           { ", " * }
1991           { "DOI: " * }
1992           if$
1993           "\doi{" s * "}" * *
1994         }
1995         if$
1996         "" 's :=
1997       }
1998       { s tmp.str * 's := }
1999       if$
2000       t #2 global.max$ substring$ 't :=
2001     }
2002     while$
2003   }
2004   'skip$
2005   if$
2006 }
2007
2008 FUNCTION {output.doi}
2009 { doi empty$ not show.doi and
2010   show.english.translation entry.lang lang.zh = and not and
2011   { new.block
2012     format.doi output
2013   }
2014   'skip$
2015   if$
2016 }
2017
2018 FUNCTION {check.electronic}

```



```

2019 { "" 'entry.url :=
2020 #0 'entry.is.electronic :=
2021     'check.doi
2022     'skip$
2023 if$
2024     'check.url
2025     'skip$
2026 if$
2027 medium empty$ not
2028     { medium "MT" = medium "DK" = or medium "CD" = or medium "OL" = or
2029         { #1 'entry.is.electronic := }
2030         'skip$
2031         if$
2032     }
2033     'skip$
2034 if$
2035 }
2036
2037 FUNCTION {format.eprint}
2038 { archivePrefix empty$ not
2039     { archivePrefix }
2040     { eprinttype empty$ not
2041         { archivePrefix }
2042         { "" }
2043     if$
2044     }
2045 if$
2046 's :=
2047 s empty$ not
2048     { s ": \eprint{" *
2049         url empty$ not
2050             { url }
2051             { "https://" s "l" change.case$ * ".org/abs/" * eprint * }
2052         if$
2053         * "}" *
2054         eprint * "}" *
2055     }
2056     { eprint }
2057 if$
2058 }
2059
2060 FUNCTION {output.eprint}
2061 { show.preprint eprint empty$ not and
2062     { new.block
2063         format.eprint output
2064     }
2065     'skip$
2066 if$
2067 }
2068
2069 FUNCTION {format.note}
2070 { note empty$ not show.note and
2071     { note }
2072     { "" }
2073 if$

```

```

2074 }
2075
2076 FUNCTION {output.translation}
2077 { show.english.translation entry.lang lang.zh = and
2078   { translation empty$ not
2079     { translation }
2080     { "[English translation missing!]" }
2081     if$
2082     " (in Chinese)" * output
2083     write$
2084     format.doi duplicate$ empty$ not
2085     { newline$
2086       write$
2087     }
2088     'pop$
2089     if$
2090     " \\" write$
2091     newline$
2092     "(" write$
2093     ""
2094     before.all 'output.state :=
2095   }
2096   'skip$
2097 if$
2098 }
2099

```

The function `empty.misc.check` complains if all six fields are empty, and if there's been no sorting or alphabetic-label complaint.

```

2100 FUNCTION {empty.misc.check}
2101 { author empty$ title empty$
2102   year empty$
2103   and and
2104   key empty$ not and
2105   { "all relevant fields are empty in " cite$ * warning$ }
2106   'skip$
2107 if$
2108 }
2109

```

B.5 Functions for all entry types

Now we define the type functions for all entry types that may appear in the .BIB file—e.g., functions like ‘article’ and ‘book’. These are the routines that actually generate the .BBL-file output for the entry. These must all precede the READ command. In addition, the style designer should have a function ‘default.type’ for unknown types. Note: The fields (within each list) are listed in order of appearance, except as described for an ‘inbook’ or a ‘proceedings’.

B.5.1 专著

```

2110 FUNCTION {monograph}
2111 { output.bibitem
2112   output.translation
2113   author empty$ not
2114     { format.authors }
2115     { editor empty$ not
2116       { format.editors }
2117       { "empty author and editor in " cite$ * warning$
2118 (*author-year)
2119       bbl.anonymous
2120 (/author-year)
2121 (*numerical)
2122       ""
2123 (/numerical)
2124     }
2125     if$
2126   }
2127   if$
2128   output
2129   year.after.author
2130   { period.after.author
2131     'new.sentence
2132     'skip$
2133     if$
2134     format.year "year" output.check
2135   }
2136   'skip$
2137   if$
2138   new.block
2139   format.series.vol.num.title "title" output.check
2140   "M" set.entry.mark
2141   format.mark "" output.after
2142   new.block
2143   format.translators output
2144   new.sentence
2145   format.edition output
2146   new.block
2147   format.address.publisher output
2148   year.after.author not
2149     { format.year "year" output.check }
2150     'skip$
2151   if$
2152   format.pages bbl.pages.colon output.after
2153   format.urldate "" output.after
2154   output.url
2155   output.doi
2156   new.block
2157   format.note output
2158   fin.entry
2159 }
2160

```

B.5.2 专著中的析出文献

An incollection is like inbook, but where there is a separate title for the referenced thing (and perhaps an editor for the whole). An incollection may CROSSREF a book.

Required: author, title, booktitle, publisher, year

Optional: editor, volume or number, series, type, chapter, pages, address, edition, month, note

```
2161 FUNCTION {incollection}
2162 { output.bibitem
2163   output.translation
2164   format.authors output
2165   author format.key output
2166   year.after.author
2167     { period.after.author
2168       'new.sentence
2169       'skip$
2170       if$
2171       format.year "year" output.check
2172     }
2173   'skip$
2174   if$
2175   new.block
2176   format.title "title" output.check
2177   "M" set.entry.mark
2178   format.mark "" output.after
2179   new.block
2180   format.translators output
2181   new.slash
2182   format.editors output
2183   new.block
2184   format.series.vol.num.booktitle "booktitle" output.check
2185   new.block
2186   format.edition output
2187   new.block
2188   format.address.publisher output
2189   year.after.author not
2190     { format.year "year" output.check }
2191   'skip$
2192   if$
2193   format.extracted.pages bbl.pages.colon output.after
2194   format.urldate "" output.after
2195   output.url
2196   output.doi
2197   new.block
2198   format.note output
2199   fin.entry
2200 }
2201
```

B.5.3 连续出版物

```
2202 FUNCTION {periodical}
2203 { output.bibitem
2204   output.translation
2205   format.authors output
2206   author format.key output
2207   year.after.author
2208     { period.after.author
2209       'new.sentence
2210       'skip$
2211       if$
2212       format.year "year" output.check
2213     }
2214     'skip$
2215   if$
2216   new.block
2217   format.title "title" output.check
2218   "J" set.entry.mark
2219   format.mark "" output.after
2220   new.block
2221   format.periodical.year.volume.number output
2222   new.block
2223   format.address.publisher output
2224   year.after.author not
2225     { format.periodical.year "year" output.check }
2226     'skip$
2227   if$
2228   format.urldate "" output.after
2229   output.url
2230   output.doi
2231   new.block
2232   format.note output
2233   fin.entry
2234 }
2235
```

B.5.4 连续出版物中的析出文献

The article function is for an article in a journal. An article may CROSSREF another article.

Required fields: author, title, journal, year

Optional fields: volume, number, pages, month, note

The other entry functions are all quite similar, so no "comment version" will be given for them.

```
2236 FUNCTION {journal.article}
2237 { output.bibitem
2238   output.translation
2239   format.authors output
2240   author format.key output
2241   year.after.author
2242     { period.after.author
2243       'new.sentence
```

```

2244         'skip$
2245         if$
2246             format.year "year" output.check
2247     }
2248     'skip$
2249     if$
2250     new.block
2251     title.in.journal
2252     { format.title "title" output.check
2253       "J" set.entry.mark
2254       format.mark "" output.after
2255       new.block
2256     }
2257     'skip$
2258     if$
2259     format.journal "journal" output.check
2260     year.after.author not
2261     { format.date "year" output.check }
2262     'skip$
2263     if$
2264     format.journal.volume output
2265     format.journal.number "" output.after
2266     format.journal.pages bbl.pages.colon output.after
2267     format.urldate "" output.after
2268     output.url
2269     output.doi
2270     new.block
2271     format.note output
2272     fin.entry
2273 }
2274

```

B.5.5 专利文献

number 域也可以用来表示专利号。

```

2275 FUNCTION {patent}
2276 { output.bibitem
2277   output.translation
2278   format.authors output
2279   author format.key output
2280   year.after.author
2281     { period.after.author
2282       'new.sentence
2283       'skip$
2284       if$
2285         format.year "year" output.check
2286     }
2287     'skip$
2288     if$
2289     new.block
2290     format.title "title" output.check
2291     "P" set.entry.mark
2292     format.mark "" output.after
2293     new.block

```

```

2294 format.date "year" output.check
2295 format.urldate "" output.after
2296 output.url
2297 output.doi
2298 new.block
2299 format.note output
2300 fin.entry
2301 }
2302

```

B.5.6 电子资源

```

2303 FUNCTION {electronic}
2304 { #1 #1 check.electronic
2305   #1 'entry.is.electronic :=
2306   #1 'is.pure.electronic :=
2307   output.bibitem
2308   output.translation
2309   format.authors output
2310   author format.key output
2311   year.after.author
2312     { period.after.author
2313       'new.sentence
2314       'skip$
2315       if$
2316       format.year "year" output.check
2317     }
2318     'skip$
2319   if$
2320   new.block
2321   format.series.vol.num.title "title" output.check
2322   "EB" set.entry.mark
2323   format.mark "" output.after
2324   new.block
2325   format.address.publisher output
2326   year.after.author not
2327     { date empty$
2328       { format.date output }
2329       'skip$
2330     if$
2331     }
2332     'skip$
2333   if$
2334   format.pages bbl.pages.colon output.after
2335   format.editdate "" output.after
2336   format.urldate "" output.after
2337   output.url
2338   output.doi
2339   new.block
2340   format.note output
2341   fin.entry
2342 }
2343

```

B.5.7 预印本

```
2344 FUNCTION {preprint}
2345 { output.bibitem
2346   output.translation
2347   author empty$ not
2348     { format.authors }
2349     { editor empty$ not
2350       { format.editors }
2351       { "empty author and editor in " cite$ * warning$
2352 (*author-year)
2353       bbl.anonymous
2354 (/author-year)
2355 (*numerical)
2356       ""
2357 (/numerical)
2358     }
2359     if$
2360   }
2361   if$
2362   output
2363   year.after.author
2364     { period.after.author
2365       'new.sentence
2366       'skip$
2367       if$
2368         format.year "year" output.check
2369       }
2370       'skip$
2371     if$
2372     new.block
2373     title.in.journal
2374       { format.series.vol.num.title "title" output.check
2375 (*2015)
2376       "A" set.entry.mark
2377 (/2015)
2378 (*!2015)
2379       "Z" set.entry.mark
2380 (/!2015)
2381       format.mark "" output.after
2382       new.block
2383     }
2384     'skip$
2385   if$
2386   format.translators output
2387   new.sentence
2388   format.edition output
2389   new.block
2390   year.after.author not
2391     { date empty$
2392       { format.date output }
2393       'skip$
2394     if$
2395   }
2396   'skip$
```



```

2397 if$
2398 format.pages bbl.pages.colon output.after
2399 format.editdate "" output.after
2400 format.urldate "" output.after
2401 output.eprint
2402 output.url
2403 new.block
2404 format.note output
2405 fin.entry
2406 }
2407

```

B.5.8 其他文献类型

A misc is something that doesn't fit elsewhere.

Required: at least one of the 'optional' fields

Optional: author, title, howpublished, month, year, note

Misc 用来自动判断类型。

```

2408 FUNCTION {misc}
2409 { get.journal.title
2410   duplicate$ empty$ not
2411   { check.arxiv.preprint
2412     'preprint
2413     'journal.article
2414     if$
2415   }
2416   { pop$
2417     booktitle empty$ not
2418     'incollection
2419     { publisher empty$ not
2420       'monograph
2421       { eprint empty$ not archivePrefix empty$ not or
2422         'preprint
2423         { entry.is.electronic
2424           'electronic
2425           {
2426             <!*2005>
2427               "Z" set.entry.mark
2428             </!2005>
2429             <*2005>
2430               "M" set.entry.mark
2431             </2005>
2432             monograph
2433           }
2434         } if$
2435       }
2436     } if$
2437   }
2438   if$
2439 }
2440 if$
2441 }
2442 if$

```

```

2443 empty.misc.check
2444 }
2445
2446 FUNCTION {archive}
2447 { "A" set.entry.mark
2448   misc
2449 }
2450
2451 FUNCTION {article} { misc }
2452

```

The book function is for a whole book. A book may CROSSREF another book.

Required fields: author or editor, title, publisher, year

Optional fields: volume or number, series, address, edition, month, note

```

2453 FUNCTION {book} { monograph }
2454

```

A booklet is a bound thing without a publisher or sponsoring institution.

Required: title

Optional: author, howpublished, address, month, year, note

```

2455 FUNCTION {booklet} { book }
2456
2457 FUNCTION {collection}
2458 { "G" set.entry.mark
2459   monograph
2460 }
2461
2462 FUNCTION {database}
2463 { "DB" set.entry.mark
2464   electronic
2465 }
2466
2467 FUNCTION {dataset}
2468 { "DS" set.entry.mark
2469   electronic
2470 }
2471

```

An inbook is a piece of a book: either a chapter and/or a page range. It may CROSSREF a book. If there's no volume field, the type field will come before number and series.

Required: author or editor, title, chapter and/or pages, publisher, year

Optional: volume or number, series, type, address, edition, month, note

inbook 类是不含 booktitle 域的，所以不应该适用于“专著中的析出文献”，而应该是专著，即 book 类。

```

2472 FUNCTION {inbook} { book }
2473

```

An inproceedings is an article in a conference proceedings, and it may CROSSREF a proceedings. If there's no address field, the month (& year) will appear just

before note.

Required: author, title, booktitle, year

Optional: editor, volume or number, series, pages, address, month, organization,

publisher, note

```
2474 FUNCTION {inproceedings}
2475 { "C" set.entry.mark
2476   incollection
2477 }
2478
```

The conference function is included for Scribe compatibility.

```
2479 FUNCTION {conference} { inproceedings }
2480
2481 FUNCTION {legislation} { archive }
2482
2483
2484 FUNCTION {map}
2485 { "CM" set.entry.mark
2486   misc
2487 }
2488
```

A manual is technical documentation.

Required: title

Optional: author, organization, address, edition, month, year, note

```
2489 FUNCTION {manual} { monograph }
2490
```

A mastersthesis is a Master's thesis.

Required: author, title, school, year

Optional: type, address, month, note

```
2491 FUNCTION {mastersthesis}
2492 { "D" set.entry.mark
2493   monograph
2494 }
2495
2496 FUNCTION {newspaper}
2497 { "N" set.entry.mark
2498   article
2499 }
2500
2501 FUNCTION {online}
2502 { "EB" set.entry.mark
2503   electronic
2504 }
2505
```

A phdthesis is like a mastersthesis.

Required: author, title, school, year

Optional: type, address, month, note

```
2506 FUNCTION {phdthesis} { mastersthesis }
2507
```

A proceedings is a conference proceedings. If there is an organization but no editor field, the organization will appear as the first optional field (we try to make the first block nonempty); if there's no address field, the month (& year) will appear just before note.

Required: title, year

Optional: editor, volume or number, series, address, month, organization, publisher, note

```
2508 FUNCTION {proceedings}
2509 { "C" set.entry.mark
2510   monograph
2511 }
2512
2513 FUNCTION {software}
2514 { "CP" set.entry.mark
2515   electronic
2516 }
2517
2518 FUNCTION {standard}
2519 { "S" set.entry.mark
2520   misc
2521 }
2522
```

A techreport is a technical report.

Required: author, title, institution, year

Optional: type, number, address, month, note

```
2523 FUNCTION {techreport}
2524 { "R" set.entry.mark
2525   misc
2526 }
2527
```

An unpublished is something that hasn't been published.

Required: author, title, note

Optional: month, year

```
2528 FUNCTION {unpublished} { misc }
2529
```

We use entry type 'misc' for an unknown type; BibTeX gives a warning.

```
2530 FUNCTION {default.type} { misc }
2531
```

B.6 Common macros

Here are macros for common things that may vary from style to style. Users are encouraged to use these macros.

Months are either written out in full or abbreviated

```
2532 MACRO {jan} {"January"}
2533
2534 MACRO {feb} {"February"}
2535
2536 MACRO {mar} {"March"}
2537
2538 MACRO {apr} {"April"}
2539
2540 MACRO {may} {"May"}
2541
2542 MACRO {jun} {"June"}
2543
2544 MACRO {jul} {"July"}
2545
2546 MACRO {aug} {"August"}
2547
2548 MACRO {sep} {"September"}
2549
2550 MACRO {oct} {"October"}
2551
2552 MACRO {nov} {"November"}
2553
2554 MACRO {dec} {"December"}
2555
```

Journals are either written out in full or abbreviated; the abbreviations are like those found in ACM publications.

To get a completely different set of abbreviations, it may be best to make a separate .bib file with nothing but those abbreviations; users could then include that file name as the first argument to the \bibliography command

```
2556 MACRO {acmcs} {"ACM Computing Surveys"}
2557
2558 MACRO {acta} {"Acta Informatica"}
2559
2560 MACRO {cacm} {"Communications of the ACM"}
2561
2562 MACRO {ibmjrd} {"IBM Journal of Research and Development"}
2563
2564 MACRO {ibmsj} {"IBM Systems Journal"}
2565
2566 MACRO {ieeese} {"IEEE Transactions on Software Engineering"}
2567
2568 MACRO {ieeetc} {"IEEE Transactions on Computers"}
2569
2570 MACRO {ieeetcad} {"IEEE Transactions on Computer-Aided Design of Integrated Circuits"}
2571 {"IEEE Transactions on Computer-Aided Design of Integrated Circuits"}
2572
2573 MACRO {ipl} {"Information Processing Letters"}
2574
2575 MACRO {jacm} {"Journal of the ACM"}
2576
2577 MACRO {jcss} {"Journal of Computer and System Sciences"}
```

```

2578
2579 MACRO {scp} {"Science of Computer Programming"}
2580
2581 MACRO {sicmp} {"SIAM Journal on Computing"}
2582
2583 MACRO {tocs} {"ACM Transactions on Computer Systems"}
2584
2585 MACRO {tods} {"ACM Transactions on Database Systems"}
2586
2587 MACRO {tog} {"ACM Transactions on Graphics"}
2588
2589 MACRO {toms} {"ACM Transactions on Mathematical Software"}
2590
2591 MACRO {toois} {"ACM Transactions on Office Information Systems"}
2592
2593 MACRO {toplas} {"ACM Transactions on Programming Languages and Systems"}
2594
2595 MACRO {tcs} {"Theoretical Computer Science"}
2596

```

B.7 Format labels

The `sortify` function converts to lower case after purifying; it's used in sorting and in computing alphabetic labels after sorting

The `chop.word(w,len,s)` function returns either `s` or, if the first `len` letters of `s` equals `w` (this comparison is done in the third line of the function's definition), it returns that part of `s` after `w`.

```

2597 FUNCTION {sortify}
2598 { purify$
2599   "l" change.case$
2600 }
2601

```

We need the `chop.word` stuff for the dubious `unsorted-list-with-labels` case.

```

2602 FUNCTION {chop.word}
2603 { 's :=
2604   'len :=
2605   s #1 len substring$ =
2606     { s len #1 + global.max$ substring$ }
2607     's
2608   if$
2609 }
2610

```

The `format.lab.names` function makes a short label by using the initials of the von and Last parts of the names (but if there are more than four names, (i.e., people) it truncates after three and adds a superscripted "+"; it also adds such a "+" if the last of multiple authors is "others"). If there is only one name, and its von and Last parts combined have just a single name-token ("Knuth" has a single token,

”Brinch Hansen” has two), we take the first three letters of the last name. The boolean `et.al.char.used` tells whether we’ve used a superscripted ”+”, so that we know whether to include a LaTeX macro for it.

```
format.lab.names(s) ==
BEGIN
  numnames := num.names$(s)
  if numnames > 1 then
    if numnames > 4 then
      namesleft := 3
    else
      namesleft := numnames
    nameptr := 1
    nameresult := ""
    while namesleft > 0
      do
        if (name_ptr = numnames) and
          format.name$(s, nameptr, "{ff }{vv }{ll}{ jj}") = "others"
        then nameresult := nameresult * "{\etalchar{+}}"
          et.al.char.used := true
        else nameresult := nameresult *
          format.name$(s, nameptr, "{v}{l}")
          nameptr := nameptr + 1
          namesleft := namesleft - 1
        od
      if numnames > 4 then
        nameresult := nameresult * "{\etalchar{+}}"
        et.al.char.used := true
      else
        t := format.name$(s, 1, "{v}{l}")
        if text.length$(t) < 2 then % there's just one name-token
          nameresult := text.prefix$(format.name$(s,1,"{ll}"),3)
        else
          nameresult := t
        fi
      fi
    return nameresult
  END
```

Exactly what fields we look at in constructing the primary part of the label depends on the entry type; this selectivity (as opposed to, say, always looking at author, then editor, then key) helps ensure that ”ignored” fields, as described in the LaTeX book, really are ignored. Note that MISC is part of the deepest ‘else’ clause in the nested part of `calc.label`; thus, any unrecognized entry type in the database is handled correctly.

There is one auxiliary function for each of the four different sequences of fields we use. The first of these functions looks at the author field, and then, if necessary, the key field. The other three functions, which might look at two fields and the key field, are similar, except that the key field takes precedence over the organization field

(for labels—not for sorting).

The `calc.label` function calculates the preliminary label of an entry, which is formed by taking three letters of information from the author or editor or key or organization field (depending on the entry type and on what's empty, but ignoring a leading "The " in the organization), and appending the last two characters (digits) of the year. It is an error if the appropriate fields among author, editor, organization, and key are missing, and we use the first three letters of the `cite$` in desperation when this happens. The resulting label has the year part, but not the name part, `purify$ed` (`purify$ing` the year allows some sorting shenanigans by the user).

This function also calculates the version of the label to be used in sorting.

The final label may need a trailing 'a', 'b', etc., to distinguish it from otherwise identical labels, but we can't calculate those "extra.label"s until after sorting.

```
calc.label ==
BEGIN
  if type$ = "book" or "inbook" then
    author.editor.key.label
  else if type$ = "proceedings" then
    editor.key.organization.label
  else if type$ = "manual" then
    author.key.organization.label
  else
    author.key.label
  fi fi fi
  label := label * substring$(purify$(field.or.null(year)), -1, 2)
    % assuming we will also sort, we calculate a sort.label
  sort.label := sortify(label), but use the last four, not two, digits
END
```

```
2611 FUNCTION {format.lab.name}
2612 { "{vv~}{ll}{, jj}{, ff}" format.names$ 't :=
2613   t "others" =
2614     { citation.et.al }
2615     { t get.str.lang 'name.lang :=
2616       name.lang lang.zh = name.lang lang.ja = or
2617         { t #1 "{ll}{ff}" format.names$ }
2618         { t #1 "{vv~}{ll}" format.names$ }
2619       if$
2620     }
2621   if$
2622 }
2623
```

第一作者姓名相同、年份相同但作者数量不同时，也需要年份标签区分。比如“王临惠 等, 2010a”和“王临惠, 2010b”，所以使用 `short.label` 存储不带“et al”的版本。

```
2624 FUNCTION {format.lab.names}
2625 { 's :=
2626   s #1 format.lab.name 'short.label :=
```



```

2627 #1 'nameptr :=
2628 s num.names$ 'numnames :=
2629 ""
2630 numnames 'namesleft :=
2631 { namesleft #0 > }
2632 { s nameptr format.lab.name citation.et.al =
2633   numnames citation.et.al.min #1 - > nameptr citation.et.al.use.first > and or
2634   { bbl.space *
2635     citation.et.al *
2636     #1 'namesleft :=
2637   }
2638   { nameptr #1 >
2639     { namesleft #1 = citation.and "" = not and
2640       { citation.and * }
2641       { ", " * }
2642       if$
2643     }
2644     'skip$
2645     if$
2646     s nameptr format.lab.name *
2647   }
2648   if$
2649   nameptr #1 + 'nameptr :=
2650   namesleft #1 - 'namesleft :=
2651 }
2652 while$
2653 }
2654
2655 FUNCTION {author.key.label}
2656 { author empty$
2657   { key empty$
2658     { cite$ #1 #3 substring$ }
2659     'key
2660     if$
2661   }
2662   { author format.lab.names }
2663   if$
2664 }
2665
2666 FUNCTION {author.editor.key.label}
2667 { author empty$
2668   { editor empty$
2669     { key empty$
2670       { cite$ #1 #3 substring$ }
2671       'key
2672       if$
2673     }
2674     { editor format.lab.names }
2675     if$
2676   }
2677   { author format.lab.names }
2678   if$
2679 }
2680
2681 FUNCTION {author.key.organization.label}

```

```

2682 { author empty$
2683   { key empty$
2684     { organization empty$
2685       { cite$ #1 #3 substring$ }
2686       { "The " #4 organization chop.word #3 text.prefix$ }
2687       if$
2688     }
2689     'key
2690     if$
2691   }
2692   { author format.lab.names }
2693   if$
2694 }
2695
2696 FUNCTION {editor.key.organization.label}
2697 { editor empty$
2698   { key empty$
2699     { organization empty$
2700       { cite$ #1 #3 substring$ }
2701       { "The " #4 organization chop.word #3 text.prefix$ }
2702       if$
2703     }
2704     'key
2705     if$
2706   }
2707   { editor format.lab.names }
2708   if$
2709 }
2710
2711 FUNCTION {calc.short.authors}
2712 { "" 'short.label :=
2713   type$ "book" =
2714   type$ "inbook" =
2715   or
2716   'author.editor.key.label
2717   { type$ "collection" =
2718     type$ "proceedings" =
2719     or
2720     { editor empty$ not
2721       'editor.key.organization.label
2722       'author.key.organization.label
2723       if$
2724     }
2725     'author.key.label
2726     if$
2727   }
2728   if$
2729   'short.list :=
2730   short.label empty$
2731   { short.list 'short.label := }
2732   'skip$
2733   if$
2734 }
2735

```

如果 `label` 中有中括号“`[`”，分别用大括号保护起来，防止 `\bibitem` 处理出错。另外为了兼容 `bibunits`，“`name(year)fullname`”的每一项都要分别保护起来，参考 [tuna/thuthesis/#630](https://tuna.thuthesis.org/#630)。

```

2736 FUNCTION {calc.label}
2737 { calc.short.authors
2738   short.list "]" contains
2739   { "{" short.list * "]" * }
2740   { short.list }
2741   if$
2742   "("
2743   *
2744   format.year duplicate$ empty$
2745   short.list key field.or.null = or
2746     { pop$ "" }
2747     'skip$
2748   if$
2749   duplicate$ "]" contains
2750   { "{" swap$ * "]" * }
2751   'skip$
2752   if$
2753   *
2754   'label :=
2755   short.label
2756   "("
2757   *
2758   format.year duplicate$ empty$
2759   short.list key field.or.null = or
2760     { pop$ "" }
2761     'skip$
2762   if$
2763   *
2764   'short.label :=
2765 }
2766

```

B.8 Sorting

When sorting, we compute the sortkey by executing “presort” on each entry. The presort key contains a number of “sortify”ed strings, concatenated with multiple blanks between them. This makes things like “brinch per” come before “brinch hansen per”.

The fields used here are: the `sort.label` for alphabetic labels (as set by `calc.label`), followed by the author names (or editor names or organization (with a leading “The” removed) or key field, depending on entry type and on what’s empty), followed by year, followed by the first bit of the title (chopping off a leading “The”, “A”, or “An”). Names are formatted: Von Last First Junior. The names within a part will be separated by a single blank (such as “brinch hansen”), two will separate the name parts themselves (except the von and last), three will separate the names, four will

separate the names from year (and from label, if alphabetic), and four will separate year from title.

The `sort.format.names` function takes an argument that should be in BibTeX name format, and returns a string containing ” ”-separated names in the format described above. The function is almost the same as `format.names`.

```

2767 (*author-year)
2768 FUNCTION {sort.language.label}
2769 { entry.lang lang.zh =
2770   { lang.zh.order }
2771   { entry.lang lang.ja =
2772     { lang.ja.order }
2773     { entry.lang lang.en =
2774       { lang.en.order }
2775       { entry.lang lang.ru =
2776         { lang.ru.order }
2777         { lang.other.order }
2778         if$
2779       }
2780     } if$
2781   }
2782 } if$
2783 }
2784 if$
2785 #64 +
2786 int.to.chr$
2787 }
2788
2789 FUNCTION {sort.format.names}
2790 { 's :=
2791   #1 'nameptr :=
2792   ""
2793   s num.names$ 'numnames :=
2794   numnames 'namesleft :=
2795   { namesleft #0 > }
2796   {
2797     s nameptr "{vv{ } }{ll{ }}{ ff{ }}{ jj{ }}" format.name$ 't :=
2798     nameptr #1 >
2799     {
2800       " " *
2801       namesleft #1 = t "others" = and
2802       { "zzzzz" * }
2803       { numnames #2 > nameptr #2 = and
2804         { "zz" * year field.or.null * " " * }
2805         'skip$
2806       } if$
2807       t sortify *
2808     }
2809   } if$
2810 }
2811 { t sortify * }
2812 if$
2813 nameptr #1 + 'nameptr :=

```

```

2814     namesleft #1 - 'namesleft :=
2815 }
2816 while$
2817 }
2818

```

The `sort.format.title` function returns the argument, but first any leading "A"s, "An"s, or "The"s are removed. The `chop.word` function uses `s`, so we need another string variable, `t`

```

2819 FUNCTION {sort.format.title}
2820 { 't :=
2821   "A " #2
2822   "An " #3
2823   "The " #4 t chop.word
2824   chop.word
2825   chop.word
2826   sortify
2827   #1 global.max$ substring$
2828 }
2829

```

The auxiliary functions here, for the `presort` function, are analogous to the ones for `calc.label`; the same comments apply, except that the organization field takes precedence here over the key field. For sorting purposes, we still remove a leading "The" from the organization field.

```

2830 FUNCTION {anonymous.sort}
2831 { entry.lang lang.zh =
2832   { "yi4 ming2" }
2833   { "anon" }
2834   if$
2835 }
2836
2837 FUNCTION {warn.empty.key}
2838 { entry.lang lang.zh =
2839   { "empty key in " cite$ * warning$ }
2840   'skip$
2841   if$
2842 }
2843
2844 FUNCTION {author.sort}
2845 { key empty$
2846   { warn.empty.key
2847     author empty$
2848     { anonymous.sort }
2849     { author sort.format.names }
2850     if$
2851   }
2852   { key }
2853   if$
2854 }
2855
2856 FUNCTION {author.editor.sort}

```

```

2857 { key empty$
2858   { warn.empty.key
2859     author empty$
2860       { editor empty$
2861         { anonymous.sort }
2862         { editor sort.format.names }
2863       if$
2864     }
2865     { author sort.format.names }
2866   if$
2867 }
2868 { key }
2869 if$
2870 }
2871
2872 FUNCTION {author.organization.sort}
2873 { key empty$
2874   { warn.empty.key
2875     author empty$
2876       { organization empty$
2877         { anonymous.sort }
2878         { "The " #4 organization chop.word sortify }
2879       if$
2880     }
2881     { author sort.format.names }
2882   if$
2883 }
2884 { key }
2885 if$
2886 }
2887
2888 FUNCTION {editor.organization.sort}
2889 { key empty$
2890   { warn.empty.key
2891     editor empty$
2892       { organization empty$
2893         { anonymous.sort }
2894         { "The " #4 organization chop.word sortify }
2895       if$
2896     }
2897     { editor sort.format.names }
2898   if$
2899 }
2900 { key }
2901 if$
2902 }
2903
2904 </author-year>

```

顺序编码制的排序要简单得多

```

2905 (*numerical)
2906 INTEGERS { seq.num }
2907
2908 FUNCTION {init.seq}
2909 { #0 'seq.num :=}

```

```

2910
2911 FUNCTION {int.to.fix}
2912 { "000000000" swap$ int.to.str$ *
2913   #-1 #10 substring$
2914 }
2915
2916 /numerical

```

There is a limit, `entry.max$`, on the length of an entry string variable (which is what its `sort.key$` is), so we take at most that many characters of the constructed key, and hope there aren't many references that match to that many characters!

```

2917 FUNCTION {presort}
2918 { set.entry.lang
2919   set.entry.numbered
2920   show.url show.doi check.electronic
2921   #0 'is.pure.electronic :=
2922   calc.label
2923   label sortify
2924   " "
2925   *
2926   *author-year
2927   sort.language.label
2928   " "
2929   *
2930   type$ "book" =
2931   type$ "inbook" =
2932   or
2933     'author.editor.sort
2934     { type$ "collection" =
2935       type$ "proceedings" =
2936       or
2937         'editor.organization.sort
2938         'author.sort
2939       if$
2940     }
2941   if$
2942   *
2943   " "
2944   *
2945   year field.or.null sortify
2946   *
2947   " "
2948   *
2949   cite$
2950   *
2951   #1 entry.max$ substring$
2952   /author-year
2953   *numerical
2954   seq.num #1 + 'seq.num :=
2955   seq.num int.to.fix
2956   /numerical
2957   'sort.label :=
2958   sort.label *
2959   #1 entry.max$ substring$

```

```

2960 'sort.key$ :=
2961 }
2962

```

Now comes the final computation for alphabetic labels, putting in the 'a's and 'b's and so forth if required. This involves two passes: a forward pass to put in the 'b's, 'c's and so on, and a backwards pass to put in the 'a's (we don't want to put in 'a's unless we know there are 'b's). We have to keep track of the longest (in width\$ terms) label, for use by the "thebibliography" environment.

```

VAR: longest.label, last.sort.label, next.extra: string
    longest.label.width, last.extra.num: integer

initialize.longest.label ==
BEGIN
    longest.label := ""
    last.sort.label := int.to.chr$(0)
    next.extra := ""
    longest.label.width := 0
    last.extra.num := 0
END

forward.pass ==
BEGIN
    if last.sort.label = sort.label then
        last.extra.num := last.extra.num + 1
        extra.label := int.to.chr$(last.extra.num)
    else
        last.extra.num := chr.to.int$("a")
        extra.label := ""
        last.sort.label := sort.label
    fi
END

reverse.pass ==
BEGIN
    if next.extra = "b" then
        extra.label := "a"
    fi
    label := label * extra.label
    if width$(label) > longest.label.width then
        longest.label := label
        longest.label.width := width$(label)
    fi
    next.extra := extra.label
END

```

```

2963 STRINGS { longest.label last.label next.extra last.extra.label }
2964
2965 INTEGERS { longest.label.width number.label }
2966
2967 FUNCTION {initialize.longest.label}
2968 { "" 'longest.label :=
2969   #0 int.to.chr$ 'last.label :=

```



```

2970 "" 'next.extra :=
2971 #0 'longest.label.width :=
2972 #0 'number.label :=
2973 "" 'last.extra.label :=
2974 }
2975
2976 FUNCTION {forward.pass}
2977 {
2978   (*author-year)
2979   last.label short.label =
2980   { "" 'extra.label :=
2981     last.extra.label text.length$ 'charptr :=
2982     { last.extra.label charptr #1 substring$ "z" =
2983       charptr #0 > and
2984     }
2985     { "a" extra.label * 'extra.label :=
2986       charptr #1 - 'charptr :=
2987     }
2988     while$
2989     charptr #0 >
2990     { last.extra.label charptr #1 substring$ chr.to.int$ #1 + int.to.chr$
2991       extra.label * 'extra.label :=
2992       last.extra.label #1 charptr #1 - substring$
2993       extra.label * 'extra.label :=
2994     }
2995     { "a" extra.label * 'extra.label := }
2996     if$
2997     extra.label 'last.extra.label :=
2998   }
2999   { "a" 'last.extra.label :=
3000     "" 'extra.label :=
3001     short.label 'last.label :=
3002   }
3003   if$
3004   (/author-year)
3005   number.label #1 + 'number.label :=
3006 }
3007
3008 FUNCTION {reverse.pass}
3009 {
3010   (*author-year)
3011   next.extra "b" =
3012   { "a" 'extra.label := }
3013   'skip$
3014   if$
3015   extra.label 'next.extra :=
3016   extra.label
3017   duplicate$ empty$
3018   'skip$
3019   { "{\natexlab{" swap$ * "}" * }
3020   if$
3021   'extra.label :=
3022   (/author-year)
3023   label extra.label * 'label :=
3024 }

```

```

3025
3026 FUNCTION {bib.sort.order}
3027 { sort.label 'sort.key$ :=
3028 }
3029

```

B.9 Write bbl file

Now we're ready to start writing the .BBL file. We begin, if necessary, with a \LaTeX macro for unnamed names in an alphabetic label; next comes stuff from the 'preamble' command in the database files. Then we give an incantation containing the command `\begin{thebibliography}{...}` where the '...' is the longest label.

We also call `init.state.consts`, for use by the output routines.

```

3030 FUNCTION {begin.bib}
3031 { preamble$ empty$
3032   'skip$
3033   { preamble$ write$ newline$ }
3034   if$
3035   "\begin{thebibliography}{\" number.label int.to.str$ * \"" *
3036   write$ newline$
3037   terms.in.macro
3038     { "\providecommand{\biband}{和}"
3039       write$ newline$
3040       "\providecommand{\bibetal}{等}"
3041       write$ newline$
3042     }
3043     'skip$
3044     if$
3045     "\providecommand{\natexlab}[1]{#1}"
3046     write$ newline$
3047     "\providecommand{\url}[1]{#1}"
3048     write$ newline$
3049     "\expandafter\ifx\csname urlstyle\endcsname\relax\else"
3050     write$ newline$
3051     " \urlstyle{same}\fi"
3052     write$ newline$
3053     "\expandafter\ifx\csname href\endcsname\relax"
3054     write$ newline$
3055     " \DeclareUrlCommand\doi{\urlstyle{rm}}"
3056     write$ newline$
3057     " \def\epri#1#2{#2}"
3058     write$ newline$
3059     "\else"
3060     write$ newline$
3061     " \def\doi#1{\href{https://doi.org/#1}{\nolinkurl{#1}}}"
3062     write$ newline$
3063     " \let\epri\href"
3064     write$ newline$
3065     "\fi"
3066     write$ newline$
3067   }
3068

```

Finally, we finish up by writing the ‘\end{thebibliography}’ command.

```
3069 FUNCTION {end.bib}
3070 { newline$
3071   "\end{thebibliography}" write$ newline$
3072 }
3073
```

B.10 Main execution

Now we read in the .BIB entries.

```
3074 READ
3075
3076 EXECUTE {init.state.consts}
3077
3078 EXECUTE {load.config}
3079
3080 < *numerical >
3081 EXECUTE {init.seq}
3082
3083 < /numerical >
3084 ITERATE {presort}
3085
```

And now we can sort

```
3086 SORT
3087
3088 EXECUTE {initialize.longest.label}
3089
3090 ITERATE {forward.pass}
3091
3092 REVERSE {reverse.pass}
3093
3094 ITERATE {bib.sort.order}
3095
3096 SORT
3097
3098 EXECUTE {begin.bib}
3099
```

Now we produce the output for all the entries

```
3100 ITERATE {call.type$}
3101
3102 EXECUTE {end.bib}
3103 < /author-year | numerical >
```