CHEMNUM

v1.0rc4   2014/03/04

numbering of chemical compounds

Clemens NIEDERBERGER

https://github.com/cgnieder/chemnum/

contact@mychemistry.eu

Table of Contents

1 License and Requirements

Permission is granted to copy, distribute and/or modify this software under the terms of the LaTeX Project Public License (LPPL), version 1.3 or later (http://www.latex-project.org/lppl.txt). The software has the status "maintained."

CHEMNUM requires the bundles l3kernel [The13a] and l3packages [The13b]. It also requires the translations package [Nie13], chemgreek from the chemmacros bundle [Nie14] and the psfrag [GC98] package.

2 News

The CHEMNUM package has been my first attempt to create a comprehensive labeling package for chemical compounds. However, it had and has more than one weakness and its code was – to be frank – a mess. Version 1 is now a complete re-write of CHEMNUM where I tried to achieve several points:

- A cleaner code internally.

- A cleaner user interface, *i. e.*, more user macros for different tasks, a unified naming of the commands and a less redundant naming of the options.

- Extended functionality such as sorting and compressing of sublabel lists and sorting and merging of main label lists.

Although the syntax seems more or less the same at first sight quite a number of changes have been made that make version 1 incompatible with version 0. While I thought a while about maintaining backwards compatibility version 0 was known to in an experimental stage where everything was allowed to be changed at any time. The same is not true for version 1.

Older versions are still available from websites such as `ctanhg.scharrer-online.de` or `bitbucket.org/cgnieder/chemnum`. You can also email me for an older version.

Many commands have got a new name! The most important ones are:

- `\cmpdref`; this is now called `\replacecmpd`.

- `\cmpdinit`; this is now called `\initcmpd`.

- `\cmpdreset`; this is now called `\resetcmpd`.

- `\cmpdsetup`; this is now called `\setchemnum`.

However, there are many more changes. Basically all options have new names and often do their thing slightly different from the way things have been before.

Please note that this overall change does not mean that version 1 is version 0 declared stable. It is very likely that version 1 will now have quite a number of bugs to be fixed and probably missing features, too. So I'd be very glad to receive feedback either on CHEMNUM's homepage `github.com/cgnieder/chemnum` or via email to `contact@mychemistry.eu`.

3 Overview over the Available Commands

This section lists all available commands with a brief description. Commands marked with * are expandable in an `\edef` like context. Most of the commands eill be explained in a later section in more detail.

`\cmpd*+[`⟨*options*⟩`]{`⟨*list of IDS*⟩`}`
   The main command for creating and refering to compound labels. This command is described in detail in section 5. For most people this will be the only command they need.

`\refcmpd[`⟨*options*⟩`]{`⟨*ID*⟩`}`

This command only refers to an already defined label but does not define a label itself. This is an alias of `\cmpd+`.

`\labelcmpd[`⟨*options*⟩`]{`⟨*ID*⟩`}`

This command only defines a new label but does not print it. This is an alias of `\cmpd*`.

* `\cmpdplain{`⟨*ID*⟩`}`

Reads a label and writes it expandably without formatting. It is not able to parse a list. Its sole purpose is usage in PDF strings (*cf.* `\texorpdfstring{`⟨*TEX*⟩`}{`⟨*PDF string*⟩`}`). This command is described in section 5.

* `\subcmpdplain{`⟨*main ID*⟩`}{`⟨*sub ID*⟩`}`

Reads a sublabel and writes it expandably without formatting. It is not able to parse a list. Its sole purpose is usage in pdfstrings (*cf.* `\texorpdfstring{`⟨*TEX*⟩`}{`⟨*PDF string*⟩`}`). This command is described in section 5.

* `\submaincmpdplain{`⟨*main ID*⟩`}{`⟨*sub ID*⟩`}`

Reads a main label and a sublabel and writes them expandably without formatting. It is not able to parse a list. Its sole purpose is usage in PDF strings (*cf.* `\texorpdfstring{`⟨*TEX*⟩`}{`⟨*PDF string*⟩`}`). This command is described in section 5.

`\initcmpd[`⟨*options*⟩`]{`⟨*list of IDs*⟩`}`

Initiate compound labels. This command can only be used in the preamble. It is desribed in section 5.

* `\cmpdproperty{`⟨*ID*⟩`}{`⟨*property*⟩`}`

Get the associated property ⟨*property*⟩ of compound ⟨*ID*⟩. This command is described in section 5.

* `\subcmpdproperty{`⟨*main ID*⟩`}{`⟨*sub ID*⟩`}{`⟨*property*⟩`}`

Get the associated property ⟨*property*⟩ of subcompound ⟨*sub ID*⟩ of compound ⟨*main ID*⟩. This command is described in section 5.

`\newcmpdcounterformat{`⟨*name*⟩`}{`⟨*command*⟩`}`

Makes the label format ⟨*name*⟩ known to CHEMNUM. ⟨*command*⟩ needs to be a command that takes an integer number as argument and should return a formatted version of it. In practice you should not need to use this command as the most common formats already are defined. This command is described in section 7.1.

`\resetcmpd[`⟨*integer*⟩`]`   Default: 1

Reset the numbering for main compound labels to start with ⟨*integer*⟩ again. This is the same as `\setcounter{cmpdmain}{`⟨*integer*⟩ − 1`}`. The command is described in section 7.2.

`\cmpdshowdef{`⟨*ID*⟩`}`

Internal command used to display ⟨*ID*⟩ of a newly defined compound label when the option `show-keys` is used. The command is described in section 12.

\cmpdshowref{⟨*ID*⟩}

Internal command used to display ⟨*ID*⟩ of a referencing compound label when the option show-keys is used. The command is described in section 12.

\subcmpdshowdef{⟨*main ID*⟩}{⟨*sub ID*⟩}

Internal command used to display ⟨*main ID*⟩ and ⟨*sub ID*⟩ of a newly defined subcompound label when the option show-keys is used. The command is described in section 12.

\subcmpdshowref{⟨*main ID*⟩}{⟨*sub ID*⟩}

Internal command used to display ⟨*main ID*⟩ and ⟨*sub ID*⟩ of a referencing subcompound label when the option show-keys is used. The command is described in section 12.

4 Numbering Compounds

4.1 Main Command

The main command of this package is this one:

\cmpd{⟨*ID*⟩}

When ⟨*ID*⟩ is used the first time, the label is created, saved (= declared) and printed. Each further use just prints the label.

```
1 Compounds \cmpd{a} and \cmpd{b} are declared and can be used any time:
2 \cmpd{a}.  No pre-declaring is necessary.  Compounds like \cmpd{c} are
3 numbered in the order they appear in the text.\par
4 Once again: \cmpd{b}, \cmpd{a}, \cmpd{c}.
```

Compounds **1** and **2** are declared and can be used any time: **1**. No pre-declaring is necessary. Compounds like **3** are numbered in the order they appear in the text.

Once again: **2**, **1**, **3**.

If it is necessary to declare a compound without printing the label it is possible with

\cmpd*{⟨*ID*⟩}

Declare the label ⟨*ID*⟩ but don't print anything.

```
1 The hidden version\cmpd*{d} declares the label but doesn't print anything.
2 The next \cmpd{e} continues to count with the next number.  With \cmpd{d}
3 the label can be used, of course.
```

> The hidden version declares the label but doesn't print anything. The next **5** continues to count with the next number. With **4** the label can be used, of course.

You can pretty much use what you like for a label name except for the separator symbols (see also section 10). Be careful with blanks though! Leading and trailing spaces are ignored, spaces at other places are not. It's probably best not to use blanks in label names at all.

```
1 \cmpd{aa},␣\cmpd{aa␣},␣\cmpd{␣aa},␣and␣\cmpd{␣aa␣}␣all␣have␣the␣same␣label.
2 Likewise␣\cmpd{a␣a},␣\cmpd{a␣a␣},␣\cmpd{␣a␣a},␣\cmpd{␣a␣a␣},␣\cmpd{a␣␣a},
3 \cmpd{a␣␣a␣},␣\cmpd{␣a␣␣a},␣and␣\cmpd{␣a␣␣a␣}.
```

6, **6**, **6**, and **6** all have the same label. Likewise **7**, **7**, **7**, **7**, **7**, **7**, **7**, and **7**.

4.2 Sublabels

If you want a label like **1a**, you need to use the following syntax:

\cmpd{⟨*main ID*⟩.⟨*sub ID*⟩}

⟨*main ID*⟩ is the main name which stays the same, ⟨*sub ID*⟩ varies. This syntax means that the point . *cannot* be a part of ⟨*main ID*⟩ or ⟨*sub ID*⟩ (except if you enclose the respective ID in braces). Instead of the point you also can use another symbol, see section 10 for details.

```
1 \cmpd{f.one} and \cmpd{f.two} are related, as are \cmpd{g.one} and
2 \cmpd{g.two}.  Of course these labels can be used again: \cmpd{g.two} and
3 \cmpd{f.one}.
```

8a and **8b** are related, as are **9a** and **9b**. Of course these labels can be used again: **9b** and **8a**.

This also works if the main name has already been used.

```
1 \cmpd{a} and its variants \cmpd{a.one} and \cmpd{a.two}
```

---

1 and its variants **1a** and **1b**

The same way the main name of combined labels can be used solely.

```
1 \cmpd{f} and \cmpd{g}                          8 and 9
```

How you can create a combined label like **8a**,**b** is explained in section 4.4.

4.3 Lists

There is actually more to the \cmpd command. It also prints lists of labels. The right description would be something like:

\cmpd{⟨*(possibly comma separated list of) label name(s)*⟩}
 Treats each entry of the list as described before.

This means that with default settings the comma can't be part of the label name unless hidden in braces. As separator another symbol can be used, too, see section 10 for details.

```
1 More than one label can be put inside \cs{cmpd}, separated by commas.  Then
2 a list like \cmpd{a, b, c, e, g.two} is printed.
```

   More than one label can be put inside \cmpd, separated by commas. Then a list like **1**, **2**, **3**, **5**, and **9b** is printed.

The Harvard comma (see section 11) in , and between **5** and **9b** suggests that there are options to customize the list, see section 8 for more on this.
   The option merge has an effect on lists: if it is set to true multiple occurences of a main label with a possibly different set of sublabels are merged into one label:

```
1 With \keyis{merge}{true} a list like \cmpd{c,g.two,a,g.{one,four}} looks
2 like \cmpd[merge=true]{c,g.two,a,g.{one,four}}.
```

   With merge = {true} a list like **1**, **3**, **9b**, and **9a,c** looks like **1**, **3**, and **9a–c**.

4.4 Lists and Ranges of Sublabels

Sometimes it can be useful to display a label with a list or a range of sublabels. Suppose you have compounds **10a**, **10b**, **10c**, **10d**, and **10e** which for example differ in their substituents. It can be useful to refer to them all at once:
   The syntax is rather intuitive – you just input a list of sublabels:

```
1 \setchemnum{compress=false}%
2 list of labels: \cmpd{q.one, q.two, q.three, q.four, q.five}\par
3 label with list of sublabels: \cmpd{q.{one,two,three,four,five}}
```

   list of labels: **10a**, **10b**, **10c**, **10d**, and **10e**
   label with list of sublabels: **10a,b,c,d,e**

Since the sublist is input with a comma in the default setting you have to put them into braces. If you add a list of sublabels to a main label they will always be printed in the order the sublabels have been declared and not in the order they're input in the list:

```
1 \setchemnum{compress=false}%
2 compare \cmpd{q.{one,two,three,four,five}}
3 with \cmpd{q.{five,four,three,two,one}} and
4 \cmpd{q.{three,four,one,five,two}}
```

compare **10a,b,c,d,e** with **10a,b,c,d,e** and **10a,b,c,d,e**

Using this syntax you also can create ranges of sublabels. For this you enable the option compress. Or rather: this is the default setting. If you don't want compressed sublabels you have to disable the option like in the previous examples.

```
1 \cmpd{q.{two,four,three}} \par          10b−d
2 \cmpd{q.{five,one,three,four}} \par      10a,c−e
3 \cmpd{q.{one,three,five,two}}            10a−c,e
```

Obviously you can't use a comma as part of a sublabel name. You can change the input marker, though. See section 6 for available options.

Sometimes it can be useful to get only the sublabel without the main label. This is achieved with the option sub-only:

```
1 % uses packages 'chemfig', 'chemformula' and 'booktabs'
2 \chemname{\chemfig{*6(=-=-(-R)=-)}}{\cmpd{benzene.{H,Me,OH,NH2}}}
3 \quad
4 \begin{tabular}{lll}
5   \toprule
6                                 & \ch{-R}   & Name \\
7   \midrule
8     \cmpd[sub-only]{benzene.H}   & \ch{-H}   & Benzene \\
9     \cmpd[sub-only]{benzene.Me}  & \ch{-CH3} & Toluene \\
10    \cmpd[sub-only]{benzene.OH}  & \ch{-OH}  & Phenol \\
11    \cmpd[sub-only]{benzene.NH2} & \ch{-NH2} & Phenylamine (Aniline) \\
12   \bottomrule
13 \end{tabular}
```
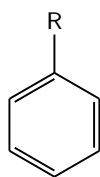
	−R	Name
a	−H	Benzene
b	−CH$_3$	Toluene
c	−OH	Phenol
d	−NH$_2$	Phenylamine (Aniline)

11a–d

4.5 Usage in Section Headings and Captions

If you use labels in section headings or captions you will want to use either `\refcmpd` or `\cmpd+` (they are completely equivalent). Otherwise the corresponding labels will be declared when the section headings appear in the table of contents or maybe the page header. This would mess up the desired order of the compound numbers.

5 Details on Compound Labels

5.1 How Things Work

When you call `\cmpd` with a new label three things happen:

- The new label gets initiated. This is nothing more than adding it to an internal list. The purpose of this is explained in section 5.3.

- The new label gets declared. This means that a number of internal commands are defined. Amongst other things they hold a number of properties associated with the corresponding label. Those properties are explained in more detail in section 5.2. The necessary information of the label are also written to the aux file.

- The label gets printed.

Since new labels are declared when `\cmpd` is first used using it in section titles that are written to the table of contents may to lead to wrong numbering. In order to avoid this compound label information is written to the aux file. The command `\refcmpd[`⟨*options*⟩`]{`⟨*ID*⟩`}` only reads those information but does not declare a label. There is also a command which does the opposite: it declares a label if it hasn't been declared before but will not print the corresponding label: `\labelcmpd[`⟨*options*⟩`]{`⟨*ID*⟩`}`. Both commands have shortcut versions: `\cmpd+` is the same as `\refcmpd`, `\cmpd*` is the same as `\labelcmpd`.

Another command available is `\cmpdplain{`⟨*ID*⟩`}`. This command is similar to `\refcmpd`. There are a few important differences, though: `\cmpdplain` does *not* take a list of labels as argument. It also is *not* able to interpret sublabels. `\cmpdplain` does not format the label with whatever format has been declared. And last but not least: it is expandable. This means it can be used to get labels in PDF bookmarks. It's equivalent `\subcmpdplain{`⟨*main ID*⟩`}{`⟨*sub ID*⟩`}` does the same for sublabels. A third sibling, `\submaincmpdplain{`⟨*main ID*⟩`}{`⟨*sub ID*⟩`}`, writes both the main and the sublabel.

I should also say a few words on lists of labels. A usage like `\cmpd{a,b,c,e}` will be printed as a *sorted* list. The order will be in the order *in which the labels have been defined*. The above usage gives **1**, **2**, **3**, and **5**. The same thing holds for the order of sublabels of a compound: the usage `\cmpd{q.{one,three,four,two,five}}` gives **10a**–**e** or **10a**,**b**,**c**,**d**,**e** (depending on the `compress` option). Be careful if you have a list with several occurences of the same main label but with different sublabels: the labels will not be sorted depending on their sublabels.

```
1 \cmpd{q.five,q.two} \par
2 \cmpd{q.two,q.five} \par
3 \cmpd[merge]{q.five,q.two} \par
4 \cmpd{q.five,b,r,q.two,a}
```

10e and **10b**

10b and **10e**

10b,**e**

1, **2**, **10e**, **10b**, and **12**

A last thing on lists: duplicate entries (*i. e.*, *exact* duplicates) will be removed.

5.2 Properties of Compound Labels

Every label has a number of properties. The first property is of course its ID which identifies the label. The other properties are:

number An internal unique number.

counter-representation The counter representation associated with the label. This is the actual label that get's printed.

pre-label-code Code to be inserted before the label is printed.

post-label-code Code to be inserted after the complete label is printed.

pre-main-label-code Code to be inserted before the *main* label is printed.

post-main-label-code Code to be inserted after the *main* label is printed.

label-format Formatting commands for the label. This is most likely something like `\bfseries`. This is the *default* format. Unlike the other properties it can be changed locally with the `format` option on a case by case basis.

The properties for a label are set the when a label is declared for the first time.

* `\cmpdproperty{`⟨*ID*⟩`}{`⟨*property*⟩`}`
  Get the associated property ⟨*property*⟩ of compound ⟨*ID*⟩. This command is expandable.

```
1 \def\expandfull{\romannumeral-'0}%
2 \def\expandtwice{\detokenize\expandafter\expandafter\expandafter}%
3 \ttfamily
4 number: \cmpdproperty{benzene}{number}\par
5 counter-representation: \cmpdproperty{benzene}{counter-representation}\par
6 pre-label-code: \cmpdproperty{benzene}{pre-label-code}\par % empty
7 post-label-code: \cmpdproperty{benzene}{post-label-code}\par % empty
```

```
8 label-format: \expandtwice{\expandfull\cmpdproperty{benzene}{label-format}}
```

```
  number:  11
  counter-representation:  11
  pre-label-code:
  post-label-code:
  label-format:  \protect \bfseries
```

Similarly a sublabel has associated properties. Additionally to the obvious ones – its ID and the ID the main label it belongs to – these are

number An internal unique number.

counter-representation The counter representation associated with the label. This is the actual label that get's printed.

* \subcmpdproperty{⟨*main ID*⟩}{⟨*sub ID*⟩}{⟨*property*⟩}
Get the associated property ⟨*property*⟩ of subcompound ⟨*sub ID*⟩ of compound ⟨*main ID*⟩. This command is expandable.

```
1 \ttfamily
2 main-compound: \subcmpdproperty{benzene}{OH}{main-compound}\par
3 number: \subcmpdproperty{benzene}{OH}{number}\par
4 counter-representation: \subcmpdproperty{benzene}{OH}{counter-representation}
```

```
  main-compound:  benzene
  number:  3
  counter-representation:  c
```

If you compile with the log = {verbose} all properties of a label are listed in the log when it is declared. This will typically look like this:

```
1 ...............................................
2 . chemnum info: defined new compound:
3 .    ID = a
4 .    internal number = 1
5 .    label = A
6 .    pre label code =
7 .    post label code =
8 .    pre main label code =
9 .    post main label code =
```

```
10 .    format = \bfseries
11 ..........................................
```

5.3 Initiating Labels

Initiating labels is not the same as declaring them although it happens simultaneously. When a label is *initiated* its ID is added to an internal list. When a label is *declared* all of its properties and associated macros are defined. Initiating can serve two purposes:

1. It can help in keeping track of defined labels; if you set the option init CHEMNUM will either issue a warning or an error (depending on the actual setting you chose) if a label is used (and hence probably declared) *that hasn't been initiated.* This can also help in detecting typos in label names.

2. Since the labels are declared in the preamble you don't need to worry about a label erroneously being declared in the table of contents. This means the variants \cmpd* and \cmpd+ shouldn't be needed.

Initiating is done via the command \initcmpd:

```
1 \initcmpd{a,b,c,d}
```

You simply use all IDs you want to use like you would use them in \cmpd. \initcmpd also has an optional argument that allows you to set options for those labels. Legal options are the same as for \cmpd.

Remember: \initcmpd will both initiate the labels *and* declare the labels!

6 Overview over the Available Options

All of the following options are either set as options to \cmpd or \initcmpd directly or via \setchemnum{⟨*options*⟩}, each time as a comma separated list of key/value pairs. Options that can only be set via \setchemnum are marked with general, those that only have an effect when used with \cmpd and friends are marked with cmpd. Those marked with both can be set either way. The options affecting the compounds are further divided in two classes: I named them global (g) and local (l). Options from the global class are set when a label is declared the first time and then are a fixed property of the corresponding label. Options from the local class can be changed at each instance of a label and will then only be active for the one instance.

A few of the options only have an effect when used with the \replacecmpd command. They are marked with replace.

general » counter-within = {⟨*counter*⟩}
    Reset the compound numbers when ⟨*counter*⟩ is stepped.

`both (g) »` `counter-format` = arabic|alph|Alph|roman|Roman|greek|Greek   Default: arabic
         The format of the number associated with the main compounds.

`both (g) »` `sub-counter-format` = arabic|alph|Alph|roman|Roman|greek|Greek   Default: alph
         The format of the number associated with the sub compounds.

`both (l) »` `compress` = <u>true</u>|false   Default: true
         If set to true a list of sublabels is compressed, *i. e.*, **10a,c,d,e** becomes **10a,c–e**.

`both (l) »` `merge` = <u>true</u>|false   Default: false
         If set to true a list of labels is merged, *i. e.*, "**1**, **10e**, and **10b**" becomes "**1** and **10b,e**".

`cmpd (g) »` `pre-label-code` = {⟨*code*⟩}   (initially empty)
         Code to be inserted before a label.

`cmpd (g) »` `post-label-code` = {⟨*code*⟩}   (initially empty)
         Code to be inserted after a label.

`both (l) »` `main-sub-sep` = {⟨*code*⟩}   Default: .
         The separator symbol that is used in \cmpd to separate the ⟨*main ID*⟩ from a ⟨*sub ID*⟩.

`both (l) »` `format` = {⟨*formatting commands*⟩}   Default: \bfseries
         The default format of the labels.

`both (l) »` `list-label-sep` = {⟨*code*⟩}   Default: ,
         The separator that is used to separate different ⟨*main IDs*⟩ in \cmpd.

`both (l) »` `sub-list-label-sep` = {⟨*code*⟩}   Default: ,
         The marker that is used to split an input list of sublabels.

`both (l) »` `list-sep-two` = {⟨*code*⟩}   Default: ␣and␣
         The output separator between labels in a list that contains of two items.

`both (l) »` `list-sep-more` = {⟨*code*⟩}   Default: ,␣
         The output separator between labels in a list that contains of more than two items.

`both (l) »` `list-sep-last-two` = {⟨*code*⟩}   Default: ,␣and␣
         The output separator between the last two labels in a list that contains of more than two items.

`cmpd (l) »` `sub-only` = <u>true</u>|false   Default: false
         If true the command \cmpd will only print sublabels but no main labels.

`cmpd (l) »` `sub-all` = <u>true</u>|false   Default: false
         If true the command \cmpd will print all sublabels belonging to the corresponding main label.

`both (l) »` `sub-list-sep-two` = {⟨*code*⟩}   Default: ,
         The output separator between labels in a sublist that contains of two items.

`both (l) »` `sub-list-sep-more` = {⟨*code*⟩}   Default: ,
         The output separator between labels in a sublist that contains of more than two items.

both (l) » sub-list-sep-last-two = {⟨*code*⟩}   Default: ,
        The output separator between the last two labels in a sublist that contains of more than two items.

both (l) » sub-list-sep-range = {⟨*code*⟩}   Default: --
        The output separator between two labels in a sublist denoting a range. This is only used when the option `compress` is active.

general » replace-auto = <u>true</u>|false   Default: true
        When set to true this adds an incremented integer to the replacement tag.

general » replace-tag = {⟨*text*⟩}   Default: TMP
        The default replacement tag.

replace » tag = {⟨*text*⟩}   Default: TMP⟨*number*⟩
        The local replacement tag.

general » replace-style = {⟨*code*⟩}   Default: \sffamily
        Additional TeX code that it placed before the \cmpd command in the replacement.

replace » style = {⟨*code*⟩}   Default: \sffamily
        Local additional TeX code that it placed before the \cmpd command in the replacement.

general » replace-pos = {⟨*TeX pos*⟩}{⟨*PS pos*⟩}   Default: bb
        Options for psfrag's \psfrag.

replace » pos = {⟨*TeX pos*⟩}{⟨*PS pos*⟩}   Default: bb
        Local options for psfrag's \psfrag.

general » init = <u>true</u>|main|false|strict|main-strict   Default: false
        Determines how labels have to be initiated. `false` means that labels are initiated when they're used the first time in the text. `true` means that labels should be initiated in the preamble with \initcmpd. `main` is the same as `true` but only for main labels. `strict` means that if an un-initiated label is used an error is thrown. `main-strict` is the same as `strict` but only for main labels.

general » log = <u>true</u>|false|silent|verbose   Default: false
        Determines how the declaration of the labels will be logged. `false` means that no information is written to the .log file. `true` means that basic information is written to the .log file when a label or a sublabel is declared. `silent` is an alias of `true`. `verbose` means that detailed information is written to the .log file when a label or a sublabel is declared.

general » show-keys = <u>true</u>|false|def|ref   Default: false
        This option will write visual hints when a label is defined (choices `true` or `def`) or when a label is referenced (choices `true` or `ref`).

7 The Counter Settings

The default setting for main labels is arabic numbering which is the most common use case for compound labels. There are however cases when you might want a different numbering. The numbering also is not reset in a document. I have heard of cases where this might be desirable, though. This section will tell you how you can achieve those things.

7.1 Change the Numbering

The counter representation used for the main and the sublabels can be changed using the following options:

general » `counter-format` = `arabic|alph|Alph|roman|Roman|greek|Greek`   Default: `arabic`
    The format of the number associated with the main compounds.

general » `sub-counter-format` = `arabic|alph|Alph|roman|Roman|greek|Greek`   Default: `alph`
    The format of the number associated with the sub compounds.

    Those options can be set globally with `\setchemnum` or localized for the single compounds.

```
1 \cmpd[counter-format=Alph]{Alpha} and
2 \cmpd[counter-format=greek]{greek}
```
**M** and **ξ**

    While it may not be necessary very often to change the default setting one could image cases where it makes sense, *e. g.*, Greek sublabels for the anomers of a carbohydrate.

```
1 % this example uses the 'chemfig' package
2 \definesubmol{r}{(-[4]H)(-[0]OH)}
3 \definesubmol{l}{(-[0]H)(-[4]HO)}
4 \labelcmpd[sub-counter-format=greek]{glucose.{alpha,beta}}
5 \labelcmpd{glucose.chain}
6 \centering
7 \schemestart
8   \small\chemfig{
9     ?(-[:-170]HO)
10    -[:-50](-[:170]HO)
11    -[:10](-[:-55,0.7]OH)
12    -[:-10](-[6,0.8]OH)
13    -[:130]O-[:-170]?(-[:150,0.7]-[2,0.7]OH)
14   }
15   \arrow(alpha--chain){<=>}
16   \small\chemfig{[6]O=^[5]-!r-!l-!r--[7]OH}
17   \arrow(--beta){<=>}
18   \small\chemfig{
19    ?(-[:-170]HO)
20    -[:-50](-[:170]HO)
21    -[:10](-[:-55,0.7]OH)
```

```
22      -[:-10](-[:10]OH)(-[6,,,,draw=none]\vphantom{OH})
23      -[:130]O-[:-170]?(-[:150,0.7]-[2,0.7]OH)
24    }
25    \arrow(@chain--chainlabel){0}[-90,.2] \cmpd{glucose.chain}
26    \arrow(@chainlabel--){0}[,2.5]         \cmpd{glucose.beta}
27    \arrow(@chainlabel--){0}[180,2.3]      \cmpd{glucose.alpha}
28  \schemestop
```


15α   **15c**   **15β**
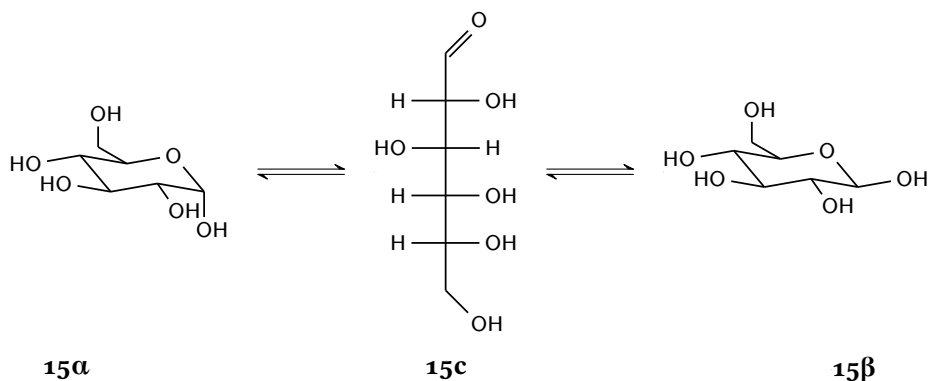
Should it ever be necessary to use another kind of counter representations than the ones already provided they can be added with this command:

\newcmpdcounterformat{⟨*name*⟩}{⟨*code*⟩}

Makes the label format ⟨*name*⟩ known to CHEMNUM. ⟨*code*⟩ needs to end with a command that takes an integer number as mandatory argument and should return a formatted version of it.

The `arabic` and `alph` counter settings for example could have been defined like this:

```
1 \newcmpdcounterformat{arabic}{\@arabic}
2 \newcmpdcounterformat{alph}  {\@alph}
```

This is actually not true: since CHEMNUM is written in expl3 the corresponding \int_to_⟨...⟩ functions have been used.

Although the name of the command suggests otherwise it can be used to overwrite the default definitions.

7.2 Reset the Numbering

There are cases when it actually might make sense to reset the counting of the compound labels. For this you can use this command:

\resetcmpd[⟨*integer*⟩]   Default: 1

Reset the numbering for main compound labels to start with ⟨*integer*⟩ again. This is the same as \setcounter{cmpdmain}{⟨*integer*⟩ − 1} which means the change is global!

Be careful, though. You might end up with the same number for different compounds:

```
1  \resetcmpd The numbering starts with 1 again: \cmpd{h,i,j}, but:
2    two compounds with the same label: \cmpd{a,h}
```

The numbering starts with 1 again: **1**, **2**, and **3**, but: two compounds with the same label: **1** and **1**

8 Formatting Labels

As you will have noticed by now labels are typeset with a bold face with the default setting of CHEMNUM. This can be changed:

both (l) » format = {⟨*formatting commands*⟩}   Default: \bfseries
      The default format of the labels.

This options works in two ways: it sets the default format that is picked up by a compound label when it is defined. When you change it later already defined labels dont change:

```
1  \setchemnum{format=\itshape}
2  \cmpd{a,b} and \cmpd{new}
```
1 and **2** and *4*

If it is applied directly to the \cmpd command it changes the formatting for this usage of the command only, regardless if the label is new or not:

```
1  \cmpd[format=\itshape]{a,b} vs
2  \cmpd{a,b}
```
1 and *2* vs **1** and **2**

There is more that you can do. Maybe you want to enclose labels in parentheses?

```
1  \cmpd[pre-label-code=(,post-label-code=)]{x, y, z.one }
```

(**5**), (**6**), and (**7a**)

Please note that these options only have an effect for *newly defined* labels since they belong to a label's properties.

Other options are the customization of the list separators:

general » list-sep-two = {⟨*code*⟩}   Default: ␣and␣
      The output separator between labels in a list that contains of two items.

general » list-sep-more = {⟨*code*⟩}   Default: ,␣
      The output separator between labels in a list that contains of more than two items.

general » `list-sep-last-two` = {⟨*code*⟩}   Default: `,␣and␣`

The output separator between the last two labels in a list that contains of more than two items.

```
1 \setchemnum{list-sep-two=;,list-sep-more=;,list-sep-last-two=;}
2 \cmpd{a, b, c, d}
```

___

1;2;3;4

In the default settings these separators are language dependent. Setting them explicitly will overwrite the language sensitivity. If you only want to adapt the separators to your language have a look at section 11.

9 Replacing Tags in EPS or PS Files

Although it is quite possible to create rection schemes within LaTeX directly – for example with the chemfig package [Tel13] – many people prefer to use a program such as CHEMDRAW for it. In order to be able to use the labels with such schemes as well the following method is usually used:

- Create the scheme and place temporary tags like `TMP1`, `TMP2` and so on where you want the compound labels to be.

- Export the scheme as EPS or PS figure where you make sure that the tags are embedded as text strings.

- Include the EPS with `\includegraphics`. Right before that use `\replacecmpd` once for every temporary tag.

Figure 1 shows a scheme with temporary tags. It is produced with the following code where the class standalone has been used to get the figure only:

```
1 % code for figure 1
2 \documentclass{standalone}
3 \usepackage{graphicx,auto-pst-pdf}
4 \begin{document}
5   \includegraphics{scheme-tmp.ps}
6 \end{document}
```

The tags now can be replaced with labels. The result is shown in figure 2.
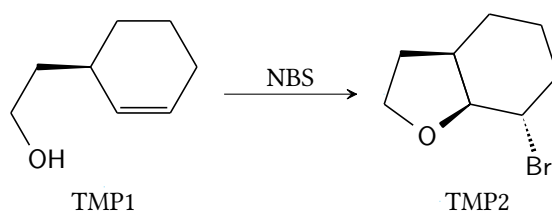
FIGURE 1: A scheme with temporary tags.

```
1  % code for figure 2
2  \documentclass{standalone}
3  \usepackage{graphicx,auto-pst-pdf,chemnum}
4  \begin{document}
5    \replacecmpd{Alc}% replaces TMP1
6    \replacecmpd{EtherBr}% replaces TMP2
7    \includegraphics{scheme-tmp.ps}
8  \end{document}
```
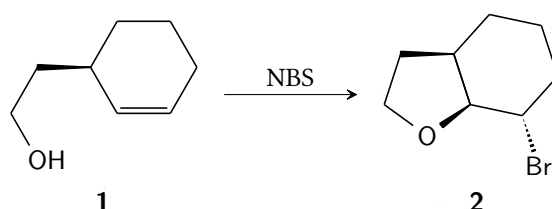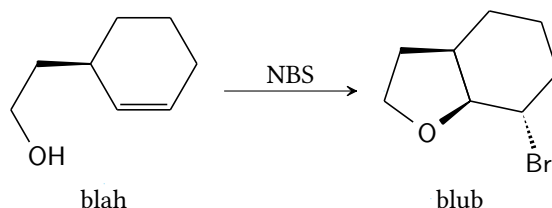

FIGURE 2: A scheme with temporary tags replaced with labels.

The replacement is done with the help of the psfrag package [GC98] and its `\psfrag` command. For details on this package and its command I refer to its documentation.

As you can see the labels are printed sans serif. This setting can of course be changed. The complete list of options is this:

general » `replace-auto` = <u>true</u>|false   Default: `true`
> When set to true this adds an incremented integer to the replacement tag.

general » `replace-tag` = {⟨*text*⟩}   Default: `TMP`
> The default replacement tag.

replace » `tag` = {⟨*text*⟩}   Default: `TMP`⟨*number*⟩
> The local replacement tag.

general » `replace-style` = {⟨*code*⟩}   Default: `\sffamily`
> Additional TEX code that it placed before the `\cmpd` command in the replacement.

replace » `style` = {⟨*code*⟩}   Default: `\sffamily`
> Local additional TEX code that it placed before the `\cmpd` command in the replacement.

general » `replace-pos` = {⟨*T_EX pos*⟩}{⟨*PS pos*⟩}   Default: bb

> Options for psfrag's `\psfrag`.

replace » `pos` = {⟨*T_EX pos*⟩}{⟨*PS pos*⟩}   Default: bb

> Local options for psfrag's `\psfrag`.

If you have a scheme with arbitrary tabs like in figure 3 you can specify the `tag` option to `\replacecmpd`. Figure 4 demonstrates this. It also demonstrates that you can of course use sublabels in the `\replacecmpd` command.

```
1 % code for figure 3
2 \documentclass{standalone}
3 \usepackage{graphicx,auto-pst-pdf}
4 \begin{document}
5   \includegraphics{scheme-bla.ps}
6 \end{document}
```


FIGURE 3: A scheme with arbitrary tags.

If you don't want to use `TMP`⟨*number*⟩ as temporary tags but for example `temp`⟨*number*⟩ you can change this with following option:

```
1 \setchemnum{replace-tag=temp}
```

The options `pos` and `replace-pos` refer to `\psfrag`'s optional arguments which determine the positioning of the T_EX box with respect to the PS box that is replaced. This is described in psfrag's documentation.

```
1 % code for figure 4
2 \documentclass{standalone}
3 \usepackage{graphicx,auto-pst-pdf,chemnum}
4 \begin{document}
5   \setchemnum{replace-style=\itshape}
6   \replacecmpd[tag=blah]{main}% replaces blah
7   \replacecmpd[tag=blub]{main.sub}% replaces blub
```

```
8    \includegraphics{scheme-bla.ps}
9  \end{document}
```
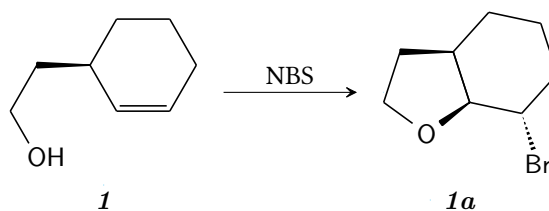

FIGURE 4: A scheme with arbitrary tags replaced with labels.

10 Changing the Input Markers

In CHEMNUM's labels there are two markers (or three, actually) that can't be part of a label name: the comma , and the dot .. You can change them with options:

general » `list-label-sep` = {⟨*token*⟩}   Default: ,

    The marker that is used to split an input list of main labels.

general » `sub-list-label-sep` = {⟨*token*⟩}   Default: ,

    The marker that is used to split an input list of sublabels.

general » `main-sub-sep` = {⟨*token*⟩}   Default: .

    The marker that divides sublabels from main labels.

```
1  \setchemnum{
2    main-sub-sep = ! ,
3    list-label-sep = ;
4  }
5  \cmpd{a; b; c; e; g!two} \par
6  \cmpd{q!one,two,three,four,five}
```

1, **2**, **3**, **5**, and **9b**

10a–**e**

11 Language Dependent Settings

A few settings of CHEMNUM depend on the language you chose with babel [Bra13] or polyglossia [Cha13]. Those regard the list separators. The language dependent strings are translated with the help of the translations [Nie13] package. This package provides the means to define translations for strings associated with identification keys. CHEMNUM defines two strings. The available languages and the corresponding translations of the two strings are listed in table 1. Note that both the comma or a leading space as well as a trailing space are part of the

translations. To make this obvious the relevant parts of the table are typeset in monotype and spaces are represented by ␣.

If you find your language missing or the translation to your language to be wrong please write me an email and I'll add your language or fix the wrong translation.

TABLE 1: Available languages

Language	`chemnum-sep-two`	`chemnum-sep-last-two`
English	`␣and␣`	`␣and␣`
American	`␣and␣`	`,␣and␣`
German	`␣und␣`	`␣und␣`
French	`␣et␣`	`␣et␣`
Spanish	`␣y␣`	`␣y␣`
Italian	`␣e␣`	`␣e␣`
Catalan	`␣i␣`	`␣i␣`
Portuguese	`␣e␣`	`␣e␣`
Dutch	`␣en␣`	`␣en␣`
Danish	`␣og␣`	`␣og␣`
Swedish	`␣och␣`	`␣och␣`
Finnish	`␣ja␣`	`␣ja␣`
Norwegian	`␣og␣`	`␣og␣`

12 Debugging Information

If you want information on the labels you have defined you can exploit the following options:

general » log = <u>true</u>|false|silent|verbose   Default: `false`

Determines how the declaration of the labels will be logged. `false` means that no information is written to the `.log` file. `true` means that basic information is written to the `.log` file when a label or a sublabel is declared. `silent` is an alias of `true`. `verbose` means that detailed information is written to the `.log` file when a label or a sublabel is declared.

general » show-keys = <u>true</u>|false|def|ref   Default: `false`

This option will write visual hints when a label is defined (choices `true` or `def`) or when a label is referenced (choices `true` or `ref`).

The option log will write information on a label to the log file when a label is defined. Depending on the choice (`true`, its alias `silent`, or `verbose`) this will be only the main information or detailed information including label properties. The following code shows an example when log = {verbose}:

```
1  .................................................
2  . chemnum info: defined new compound:
```

```
3 .    ID = a
4 .    internal number = 1
5 .    label = 1
6 .    pre label code = (
7 .    post label code = )
8 .    post main label code =
9 .    format = \bfseries
10 ...............................................
```

The `show-keys` writes some visual information to the document itself:

```
1 \setchemnum{show-keys}
2 \cmpd{a} and a bit later \cmpd{b}.
```

1 and a bit later **2**.

The last example shows the information when a label is referenced. If a label is newly declared information is written to the margin like for this label with the ID showkey: **8** that again is referencend here: **8**.

showkey

The option activates four commands:

`\cmpdshowdef{`⟨*ID*⟩`}`
Internal command used to display ⟨*ID*⟩ of a newly defined compound label when the option `show-keys` is used. The command is described in section 12.

`\cmpdshowref{`⟨*ID*⟩`}`
Internal command used to display ⟨*ID*⟩ of a referencing compound label when the option `show-keys` is used. The command is described in section 12.

`\subcmpdshowdef{`⟨*main ID*⟩`}{`⟨*sub ID*⟩`}`
Internal command used to display ⟨*main ID*⟩ and ⟨*sub ID*⟩ of a newly defined subcompound label when the option `show-keys` is used. The command is described in section 12.

`\subcmpdshowref{`⟨*main ID*⟩`}{`⟨*sub ID*⟩`}`
Internal command used to display ⟨*main ID*⟩ and ⟨*sub ID*⟩ of a referencing subcompound label when the option `show-keys` is used. The command is described in section 12.

This means you can customize the appearance of the information by redefining those commands. The ones that show the definition use a `\marginpar` in their default definition. This may cause them to disappear if issued somewhere `\marginpar` cannot be used. The following shows an equivalent definition with `\marginnote` from the marginnote package [Koh12]. (This definition has another drawback: it several `\marginnote`s can print over one another if issued in the same line.)

```
1 \renewcommand*\cmpdshowdef[1]{% /needs/ one mandatory argument
2   \marginnote{\fbox{\normalfont\ttfamily#1}}}
3 \renewcommand*\subcmpdshowdef[2]{% /needs/ two mandatory arguments
4   \marginnote{\fbox{\normalfont\ttfamily#2 (#1)}}}
5 a\cmpdshowdef{foo}\par
6 b\cmpdshowref{foo}\par
7 c\subcmpdshowdef{foo}{bar}\par
8 d\subcmpdshowref{foo}{bar}
```

a
b|foo

c
d|bar

foo

bar (foo)

Actually there are two other commands you could redefine – all four of the above commands are defined in terms of them:

```
1 \NewDocumentCommand\cmpdshowdef{m}{\chemnumshowdef{#1}}
2 \NewDocumentCommand\cmpdshowref{m}{\chemnumshowref{#1}}
3 \NewDocumentCommand\subcmpdshowdef{mm}{\chemnumshowdef{#2 (#1)}}
4 \NewDocumentCommand\subcmpdshowref{mm}{\chemnumshowdef{#2}}
```

References

[Bra13]   Johannes BRAAMS, current maintainer: Javier BEZOS.
            babel. version 3.9f, May 16, 2013.
            URL: http://mirror.ctan.org/macros/latex/required/babel/.

[Cha13]   François CHARETTE, current maintainer: Arthur REUTENAUER.
            polyglossia. version 1.33.4, June 27, 2013.
            URL: http://mirror.ctan.org/macros/latex/contrib/polyglossia/.

[GC98]   Michael C. GRANT and David CARLISLE. psfrag. version 3, Apr. 11, 1998.
            URL: http://mirror.ctan.org/graphics/psfrag/.

[Koh12]   Markus KOHM. marginnote. version 1.1i, Mar. 29, 2012.
            URL: http://mirror.ctan.org/macros/latex/contrib/marginnote/.

[Nie13]   Clemens NIEDERBERGER. translations. version 1.1a, Sept. 30, 2013.
            URL: http://mirror.ctan.org/macros/latex/contrib/translations/.

[Nie14]   Clemens NIEDERBERGER. chemmacros. version 4.3, Jan. 24, 2014.
            URL: http://mirror.ctan.org/macros/latex/contrib/chemmacros/.

[Tel13]   Christian TELLECHEA. chemfig. version 1.0h, Nov. 28, 2013.
            URL: http://mirror.ctan.org/macros/generic/chemfig/.

References

[The13a]   THE LATEX3 PROJECT TEAM. l3kernel. version SVN 4582, July 28, 2013.
            URL: http://mirror.ctan.org/macros/latex/contrib/l3kernel/.

[The13b]   THE LATEX3 PROJECT TEAM. l3packages. version SVN 4582, July 28, 2013.
            URL: http://mirror.ctan.org/macros/latex/contrib/l3packages/.

Index