Table of Contents

I. Preliminaries 2

1. Licence, Requirements and README 2

2. Motivation and Background 3

3. The Structure of CHEMMACROS 3
   3.1. General Structure ........ 3
   3.2. Using CHEMMACROS’ Options 4
   3.3. Support Package CHEMFORMULA .............. 5
   3.4. Upgrading from version < 5.0 5
   3.5. Compatibility Mode ....... 6
      3.5.1. For Users ........ 6
      3.5.2. For Module Writers .... 7

4. General Options 8

II. The Preloaded Modules 8

5. User Modules 8
   5.1. The acid-base Module .... 8
   5.2. The charges Module .... 10
      5.2.1. Charge Symbols .... 10
      5.2.2. Ion Charges .... 11
      5.2.3. Partial Charges and Similar Stuff .... 11

6. Internal Modules 24
   6.1. The base Module .... 24
   6.2. The chemformula Module .... 27
      6.2.1. For Users .... 27
      6.2.2. For Module Writers .... 29
   6.3. The errorcheck Module .... 29
   6.4. The greek Module .... 29
Part I.

Preliminaries

1. Licence, Requirements and README

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

CHEMMACROS loads the packages expl3 [L3Pa] and xparse [L3Pb]. Depending on your usage other packages will be loaded. They are mentioned when the corresponding module using the package is described.
2. Motivation and Background

This package grew from a small collection of personal helper macros back in 2010 into a rather big package supporting various different chemical typesetting tasks. I hope I have achieved the following points with this package:

- Intuitive usage as far as the syntax of the commands is concerned.

- A comprehensive set of macros! If there are any needs you might have with respect to typesetting of chemistry which is not supported by this package\(^1\) then let me know so \texttt{CHEMMACROS} can be extended.

- The commands shall not only make typesetting easier and faster but also the document source more readable with respect to semantics (\texttt{o-dichlorobenzene} is easier to read and understand than \texttt{o-dichlorobenzene}; the first variant in my opinion also is more in the spirit of \LaTeX.\)

- As much customizability as I could think of so every user can adapt the commands to his or her own wishes. Every now and then users have wishes which can’t be solved with the available options. Almost always I’ll add options then. If you find something please contact me, see section B starting on page 66.

- Default settings that are compliant with the recommendations of the \textsc{International Union of Pure and Applied Chemistry (IUPAC)}. Especially the last point in the past needed some pushing from users to get things right in many places. If you find anything not compliant with \textsc{IUPAC} recommendations please contact me, see section B starting on page 66. Don’t forget to add references for the corresponding \textsc{IUPAC} recommendation.

3. The Structure of \texttt{CHEMMACROS}

3.1. General Structure

Since version 5.0 the \texttt{CHEMMACROS} package has a strictly modular structure. On the one hand this eases maintenance but it will also allow for easy and quick extension in the future. In a way it is a logical consequence from \texttt{CHEMMACROS}' history: since version 2.0, \textit{i.e.}, since the fall of 2011 \texttt{CHEMMACROS} already had modular options.

The different modules of \texttt{CHEMMACROS} are divided into two groups:

1. Internal modules which provide underlying functionality or basic functionality which is not of direct interest from a user perspective but might be if you plan to write a module yourself (see section A for details).

2. User modules which provide all the stuff for typesetting.

Both groups each are subdivided into two groups: preloaded modules and modules which have to be loaded by the programmer (internal modules) or by the document author (user modules). Those modules are described in parts II (preloaded modules) and III (additional modules) of this manual.

The above means that not all functionality is available per default. If you want to load \textit{all} modules no matter what then you can say:

\footnote{1. Not including needs already solved by other packages such as chemnum or chemfig.}
or

\usechemmodule{all}

which will load all modules which are part of \texttt{CHEMMACROS} (also see section 7.1 starting on page 34). Own modules (see section A starting on page 65) are \textit{not} loaded through this, though, and still have to be loaded additionally.

In part II starting on page 8 the preloaded modules are described, first the user modules then the internal ones, in part III starting on page 34 the other available modules are described, again the user modules first. In each section the modules are described in an alphabetical order.

### 3.2. Using \texttt{CHEMMACROS’} Options

Prior to v5.0 \texttt{CHEMMACROS} had quite a number of package options. \texttt{CHEMMACROS} v5.0 or higher has none! All of \texttt{CHEMMACROS’} options are set using the command

\begin{verbatim}
\chemsetup{(module)}{(option list)}
\end{verbatim}

\texttt{CHEMMACROS’} setup command.

When an option is described then in the left margin the module the option belongs to is denoted. This looks something like this:

\begin{verbatim}
module » option = {\{(value)\}}
\end{verbatim}

\noindent (initially empty)

\textbf{Description of option.} The module is printed in the left margin. The default value to the right is the setting the option has when \texttt{CHEMMACROS} is loaded. This can be an explicit setting but the option can also be empty.

\begin{verbatim}
module » choice-option = list|of|choices
\end{verbatim}

\noindent \textbf{Default: list}

\textbf{Description of choice-option.} A choice option can only be used with a predefined list of values. If one of the values is underlined it means that the option can be used without value in which case the underlined value is chosen. If no value is underlined then a value \textit{has} to be given by the user.

\begin{verbatim}
module » boolean-option = true|false
\end{verbatim}

\noindent \textbf{Default: true}

\textbf{Description of boolean-option.} A boolean option is a choice option with exactly the two values \texttt{true} and \texttt{false}. If the option is called without value then the underlined value is chosen (which is always \texttt{true} for a boolean option).

An option or list of options belonging to a module \texttt{module} can be set in two ways:

\begin{verbatim}
% first possibility:
\chemsetup[module]{
  option1 = value ,
  option2 = value
}
% second possibility:
\chemsetup{
  module/option1 = value ,
\end{verbatim}
The second way allows to set options belonging to different modules with one call of \chemsetup.

### 3.3. Support Package \texttt{CHEMFORMULA}

\texttt{CHEMFORMULA} provides means of typesetting chemical formulas and reactions. You will see its macros \texttt{ch} and \texttt{chcpd} every now and then in this manual. When using \texttt{CHEMMACROS} you can consider the \texttt{CHEMFORMULA} package [Nie19] to be loaded as \texttt{CHEMMACROS} makes use of it in various places. \texttt{CHEMMACROS} and \texttt{CHEMFORMULA} are tightly intertwined. Nevertheless you should be able to use the mhchem [Hen18] package with \texttt{CHEMMACROS}. Please see section 6.2.1 starting on page 27 for details and caveats. The recommendation is to use \texttt{CHEMFORMULA}.

A historical note: \texttt{CHEMFORMULA} started as a part of \texttt{CHEMMACROS} in January 2012. Since July 2013 it is a completely independent package – from \texttt{CHEMFORMULA}'s point of view. It is maintained independently and has a manual of its own.

### 3.4. Upgrading from version \texttt{< 5.0}

People upgrading from versions \texttt{< 5.0} will find that almost everything they know from earlier versions is the same in versions \texttt{\geq 5.0}. But there are important and breaking differences:

- \texttt{CHEMMACROS} has no package options any more, all options are set via \texttt{\chemsetup}, also see section 3.2 on the previous page.

- Not all of the features are available per default any more, for some the corresponding module has to be loaded explicitly, see section 4. If suddenly some commands or environments seem to be undefined this is the most likely reason.

- Some option modules have been renamed (\textit{e.g.}, \texttt{iupac} is now \texttt{nomenclature}). If you experience strange errors when you upgrade your document this is the most likely source.

- The command family \texttt{\NewChem...}, \texttt{\RenewChem...} and \texttt{\DeclareChem...} has a new member \texttt{\ProvideChem....}.

- In \texttt{iupac} the macro \texttt{-} no longer gives a dash with breaking point. Instead \texttt{-} can be used directly.\footnote{\texttt{-} is provided up to and including v5.2 but is dropped in higher versions.}

- The macro \texttt{\ox} has another default behaviour (\texttt{pos = {super}}) and the starred version has another effect (\texttt{pos = {top}}) than the same macro in earlier versions. Now the default behaviour follows \texttt{IUPAC} recommendations. A second change is that the atom is now treated as a \texttt{CHEMFORMULA} input (this depends on the setting of the \texttt{formula} option, see 6.2 starting on page 27).

- The syntax of \texttt{\NewChemReaction} and friends is now different from what it used to be. If you have defined your own reaction environments you need to adapt!

- \texttt{CHEMMACROS} offers a macro \texttt{\state} which is similar to but different from the earlier macro \texttt{\State}. \texttt{\State} is deprecated. There are also differences in the syntax of \texttt{\enthalpy} vs. the earlier \texttt{\Enthalpy}, \texttt{\entropy} vs. \texttt{\Entropy} and \texttt{\gibbs} vs. \texttt{\Gibbs}. The uppercase versions are deprecated. The macro \texttt{\NewChemState} also has a different syntax now.
• At various places in the code improvements and fixes have been made, too many to list them here. You should keep an open eye and first of all read the manual closely.

3. The Structure of CHEMMACROS

3.5. Compatibility Mode

3.5.1. For Users

It is actually not true that CHEMMACROS has no package options any more. It has one very important package option:

\texttt{compatibility = (version)|newest|latest}

Default: 5.11a

Let’s you specify the version of CHEMMACROS you want to use. Any version earlier than 5.0 will load v4.7. i.e., the last version before CHEMMACROS got its modular structure.\footnote{Mostly: the loaded v4.7 has got a few fixes} Not using the option will always load the newest version.

Both values newest and latest will choose the latest version available.

In your document you can check for the compatibility mode. For the following functions it is important to understand the rules: \texttt{greater} means \texttt{newer}. The version number is \texttt{not} a usual decimal number! The syntax for \texttt{compatibility} is \texttt{(major).(minor)(optional subrelease)}. This means that a version 5.11 is \texttt{newer} than a version 5.7! In the same way \texttt{less} means \texttt{older}. As last example: 5.10 is \texttt{newer} (greater) than 5.1. \texttt{(optional subrelease)} is a lowercase letter which might or might not be present in a version number. Version 5.7b is newer than 5.7a which in turn is newer than 5.7.

\begin{itemize}
  \item \texttt{\IfChemCompabilityT{(\texttt{comp})}{(\texttt{version})}{\texttt{true}}{\texttt{false}}}
    \begin{itemize}
      \item Checks the value given through the option \texttt{compatibility} against \texttt{(version)} using \texttt{compatibility} and either leaves \texttt{true} or \texttt{false} in the input stream. \texttt{compatibility} can be one of \texttt{<, <=, =, >= or >}.  
    \end{itemize}
  \item \texttt{\IfChemCompabilityT{(\texttt{comp})}{(\texttt{version})}{\texttt{true}}}
    \begin{itemize}
      \item Checks the value given through the option \texttt{compatibility} against \texttt{(version)} using \texttt{compatibility} and leaves \texttt{true} in the input stream if the check is logically true. \texttt{compatibility} can be one of \texttt{<, <=, =, >= or >}.  
    \end{itemize}
  \item \texttt{\IfChemCompabilityF{(\texttt{comp})}{(\texttt{version})}{\texttt{true}}}{\texttt{false}}
    \begin{itemize}
      \item Checks the value given through the option \texttt{compatibility} against \texttt{(version)} using \texttt{compatibility} and leaves \texttt{false} in the input stream if the check is logically false. \texttt{compatibility} can be one of \texttt{<, <=, =, >= or >}.  
    \end{itemize}
  \end{itemize}

A possible usage:

\begin{itemize}
  \item \texttt{\IfChemCompabilityT{>=}{5.0}{\usechemmodule{all}}}
    \begin{itemize}
      \item Loading CHEMMACROS with \texttt{compatibility = {4.7}} also allows to use the package options from that version:
    \end{itemize}
  \item \texttt{\usepackage[compatibility=4.7,language=german]{chemmacros}}
  \end{itemize}
3. The Structure of \texttt{CHEMMACROS}

\subsubsection{For Module Writers}

For future versions the aim is not to make such breaking changes again. While we never know what the future actually will bring \texttt{CHEMMACROS} now has the tools for tying code to a version number:

\begin{verbatim}
\chemmacros_if_compatibility:nnTF {(comp)} {(version)} {(true)} {(false)}
\end{verbatim}

\begin{verbatim}
expl3 version of \IfChemCompatibilityTF.
\end{verbatim}

In modules one can try adding code for a certain version or range of versions:

\begin{verbatim}
\ChemCompatibility{(version)}{code}\EndChemCompatibility
Leaves \texttt{code} in the input stream if the compatibility version \texttt{x} given by \texttt{compatibility} matches \texttt{version} (\texttt{x} = \texttt{version}).
\end{verbatim}

\begin{verbatim}
\ChemCompatibilityFrom{(version)}{code}\EndChemCompatibility
Leaves \texttt{code} in the input stream if the compatibility version \texttt{x} given by \texttt{compatibility} matches \texttt{version} or newer. This means \texttt{version} is the oldest version allowed (\texttt{x} \geq \texttt{version}).
\end{verbatim}

\begin{verbatim}
\ChemCompatibilityTo{(version)}{code}\EndChemCompatibility
Leaves \texttt{code} in the input stream if the compatibility version \texttt{x} given by \texttt{compatibility} matches any version up to and excluding \texttt{version}. This means \texttt{version} is the oldest version not allowed (\texttt{x} < \texttt{version}).
\end{verbatim}

\begin{verbatim}
\ChemCompatibilityBetween{(version 1)}{(version 2)}{code}\EndChemCompatibility
Leaves \texttt{code} in the input stream if the compatibility version \texttt{x} given by \texttt{compatibility} lies between and including \texttt{version 1} up to and excluding \texttt{version 2} (\texttt{version 1} \leq \texttt{x} < \texttt{version 2}).
\end{verbatim}

\EndChemCompatibility

This macro must end each of the \texttt{ChemCompatibility...} macros.

You may refer to the current version of \texttt{CHEMMACROS} with the following tokenlists:

\begin{verbatim}
\c_chemmacros_date_tl
The current release date: “2020/11/21”.
\end{verbatim}

\begin{verbatim}
\c_chemmacros_version_major_number_tl
The current major version: “5”.
\end{verbatim}

\begin{verbatim}
\c_chemmacros_version_minor_number_tl
The current minor version: “11”.
\end{verbatim}

\begin{verbatim}
\c_chemmacros_version_number_tl
The current version number: “5.11”.
\end{verbatim}

\begin{verbatim}
\c_chemmacros_version_subrelease_tl
The current sub-release: “a”.
\end{verbatim}

\begin{verbatim}
\c_chemmacros_version_tl
The current version: “5.11a”.
\end{verbatim}

\begin{verbatim}
\c_chemmacros_info_tl
The package information: “comprehensive support for typesetting chemistry documents”.
\end{verbatim}
4. General Options

CHEMMACROS has some core options which don’t belong to any of the modules described in parts II and III. Those options have no module denoted in the left margin next to their descriptions and are also set without specifying a module:

\begin{verbatim}
\chemsetup{
  option1 = value ,
  option2 = value
}
\end{verbatim}

Two of those options are explained now:

\texttt{modules = \{\textit{comma separated list of module names}\}} (initially empty)
With this option you can specify which modules you want to load. Alternatively you can use \texttt{\usechemmodule{\{\textit{comma separated list of module names}\}}}.

\texttt{greek = \{\textit{mapping}\}} (initially empty)
Explicitly specify which mapping should be used by the CHEMGREEK package [Nie16a]. For details about what this means please refer to section 6.4 starting on page 29.

Some internal modules may also define core options, e.g., the \texttt{lang} module, see section 6.5 starting on page 30.

Part II.
The Preloaded Modules

5. User Modules

5.1. The acid-base Module

Easy representation of p\(H\), p\(K_a\)...

\begin{verbatim}
\pH
pH
\pOH
pOH
\Ka
\textit{K}_a, depends on language settings, see section 6.5 starting on page 30. The translations can be adapted.
\Kb
\textit{K}_b
\Kw
\textit{K}_w
\end{verbatim}
\pKa\{(num)\} \\
\pKb\{(num)\} \\
\p\{(anything)\}  \\
e.g. \p\{\KW\} \pKw

\[ \Ka \ Kb \ \pKa\{1\} \ \pKb\{1\} \]  \\
\Ka \ Kb \ pKa \ pKa\{1\} \ pKb \ pkb \ pkb\{1\}

The operator \( p \) […] shall be printed in Roman type.

The IUPAC Green Book \[\text{Coh+08, p. 103}\]

There is one option which changes the style the \( p \) is typeset, other options allow to change the subscript of the constants:

**acid-base » p-style =** \( \text{italics|slanted|upright} \)  \\
Set the style of the \( p \) operator.

**acid-base » K-acid =** \( \{(text)\} \)  \\
The subscript to \( \Ka \) and \( \pKa \).

**acid-base » K-base =** \( \{(text)\} \)  \\
The subscript to \( \Kb \) and \( \pKb \).

**acid-base » K-water =** \( \{(text)\} \)  \\
The subscript to \( \Kw \).

\textbf{acid-base » eq-constant =} \( \{(text)\} \)  \\
The symbol of the constants.

\textit{Introduced in version 5.4}  \\
(2016/02/10)

\% this works only in the preamble:  \\
\% \DeclareTranslation{English}{K-acid}{\textthrm(A)}\% use your language here

As you can see the default subscripts of \( \Ka, \pKa \) and \( \Kb \) are lowercase letters. The literature is inconclusive about if this is the right way or if uppercase letters should be preferred. In textbooks the uppercase variant usually seems to be used while journals seem to prefer the lowercase variant. CHEMMCROS’ default follows the usage in \textit{The IUPAC Green Book} \[\text{Coh+08}\]. If you want to change this you have two possibilities:
The constants $K_a$, $K_b$, and $K_w$ were defined using the following commands:

\NewChemEqConstant{cs}{name}{subscript}

Define the constant \emph{cs} with the name \emph{name} and the subscript \emph{subscript}. This also defines the default translation with the key \emph{name} using \emph{subscript} as fallback translation (see section 6.5 starting on page 30 for details). It also defines the option \emph{name} for setting the subscript.

\RenewChemEqConstant{cs}{name}{default appearance}

The same as \NewChemEqConstant but renews an existing command.

\DeclareChemEqConstant{cs}{name}{default appearance}

The same as \NewChemEqConstant but overwrites existing commands.

\ProvideChemEqConstant{cs}{name}{default appearance}

The same as \NewChemEqConstant but doesn’t throw an error if \emph{cs} already exists.

This is how $\text{pK}_A$ is defined:

\[ \text{pK}_A \]

5.2. The charges Module

The charges module loads the module chemformula.

5.2.1. Charge Symbols

\fplus

\fplus

⊕ formal positive charge

\fminus

\fminus

⊕ formal negative charge

\scrp

\scrp

+ scriptstyle positive charge (e.g., for usage in chemfig’s \text{T}el\text{19} formulas).

\scrm

\scrm

− scriptstyle negative charge (e.g., for usage in chemfig’s formulas).

\fscrp

\fscrp

⊕ scriptstyle formal positive charge (e.g., for usage in chemfig’s formulas).

\fscrm

\fscrm

⊕ scriptstyle formal negative charge (e.g., for usage in chemfig’s formulas).

\fsscrp

\fsscrp

⊕ scriptscriptstyle formal positive charge (e.g., for usage in chemfig’s formulas).
5. User Modules

\fsscrm
- scriptscriptstyle formal negative charge (e.g., for usage in chemfig’s formulas).

5.2.2. Ion Charges

Simple displaying of (real) charges. It is worth noting that these commands really are relicts from a time when \textsc{chemmacros} tried hard to be compliant with \textsc{mhchem} and \textsc{chemformula} didn’t exist, yet. \textbf{They are still provided for backwards compatibility but my recommendation is to use \texttt{ch} (see the documentation of the \texttt{chemformula} package [Nie19]) and forget about these commands:}

\pch[\texttt{(number)}]
positive charge

\mch[\texttt{(number)}]
negative charge

\fpch[\texttt{(number)}]
formal positive charge

\fmch[\texttt{(number)}]
formal negative charge

\begin{verbatim}
A\pch\ B\mch[3] C\fpch[2] D\fmch
A^+ B^{3-} C^{2\\circ} D^{\\circ}
\end{verbatim}

5.2.3. Partial Charges and Similar Stuff

The next ones probably are seldomly needed but nevertheless useful:

\delp
- partial positive charge

\delm
- partial negative charge

\fdelp
- partial formal positive charge

\fdelm
- partial formal negative charge

These macros for example can be used with the \texttt{\ox} command (see section 7.1 starting on page 46) or with the chemfig package:

\begin{verbatim}
\chemsetup{
  charges/circled = all,
  redox/parse = false,
  redox/pos = top
}
\ch{\oxy{\delp,H} -{} \oxy{\delm,Cl}} hspace*{1cm}
\chemfig{\chemabove[3pt]{\lewis{246,Br}}{\delm}-\chemabove[3pt]{H}{\delp}}
\end{verbatim}
5. User Modules

5.2.4. Charge Options

\textbf{charges} \texttt{\textbackslash circled} = \texttt{formal|all|none} \hspace{1cm} \text{Default: formal}

\textit{CHEMMACROS} uses two different kinds of charges which indicate the usage of real (+/−) and formal (⊕/⊖) charges. The option \texttt{formal} distinguishes between them, option \texttt{none} displays them all without circle, option \texttt{all} displays all circles.

\textbf{charges} \texttt{\textbackslash circletype} = \texttt{chem|math} \hspace{1cm} \text{Default: chem}

This option switches between two kinds of circled charge symbols: \texttt{\fplus \fminus} (chem) and \texttt{$\oplus \ominus$} (math).

\textbf{partial-format} = \texttt{(\textbackslash B\textbackslash i\textbackslash P\textbackslash X code}{}) \hspace{1cm} \text{charges} \texttt{\textbackslash ProvideChemPartialCharge}{}} \hspace{1cm} \text{Default: \texttt{\textbackslash tiny}}

\textbf{Code which formats the macros defined with \textbackslash NewChemPartialCharge (see section 5.2.5).}

5.2.5. Own Charge Macros

Just in case the existing macros don’t fit you needs there are commands for defining new ones or modifying the existing ones. These commands define macros like those described in section 5.2.2 on the previous page.

\textbf{\textbackslash NewChemCharge}{\langle cs\rangle}{\langle charge symbol\rangle}
\textbf{\textbackslash RenewChemCharge}{\langle cs\rangle}{\langle charge symbol\rangle}
\textbf{\textbackslash DeclareChemCharge}{\langle cs\rangle}{\langle charge symbol\rangle}
\textbf{\textbackslash ProvideChemCharge}{\langle cs\rangle}{\langle charge symbol\rangle}

An example of usage is the definition of the existing ion charge macros:

\begin{verbatim}
 1. \textbackslash NewChemCharge\fplus{fplus}
 2. \textbackslash NewChemCharge\fminus{fminus}
\end{verbatim}

These commands define macros like those described in section 5.2.3 on the preceding page.

\textbf{\textbackslash NewChemPartialCharge}{\langle cs\rangle}{\langle charge symbol\rangle}
\textbf{\textbackslash RenewChemPartialCharge}{\langle cs\rangle}{\langle charge symbol\rangle}
\textbf{\textbackslash DeclareChemPartialCharge}{\langle cs\rangle}{\langle charge symbol\rangle}
\textbf{\textbackslash ProvideChemPartialCharge}{\langle cs\rangle}{\langle charge symbol\rangle}

\textbf{An example of usage is the definition of the existing ion charge macros:}

\begin{verbatim}
 1. \textbackslash NewChemPartialCharge\fplus{fplus}
 2. \textbackslash NewChemPartialCharge\fminus{fminus}
\end{verbatim}

\textbf{These commands define macros like those described in section 5.2.3 on the preceding page.}
5. User Modules

An example of usage is the definition of the existing partial charge macros:

\begin{verbatim}
\NewChemPartialCharge\fdelp{\fplus}
\NewChemPartialCharge\fdelm{\fminus}
\end{verbatim}

5.3. The nomenclature Module

The nomenclature module loads the tikz module. It also loads the package scrlfile which is part of the KOMA-Script bundle [Koh19].

5.3.1. The \texttt{\texttt{iupac}} Command

Similar to the bpchem package [Ped17] CHEMMACROS provides a command\textsuperscript{4} for typesetting IUPAC names. Why is that useful? IUPAC names can get very long. So long indeed that they span over more than two lines, especially in two-column documents. This means they must be allowed to be broken more than one time. This is what the following command does.

\begin{quote}
\texttt{\texttt{iupac}{\{IUPAC name\}}}
\end{quote}

Inside this command use | indicate a breaking point ^ as a shortcut for \texttt{textsuperscript}. - (and ) allow words to be broken while still allow the rest of word to be hyphenated, likewise [ and ].

\begin{verbatim}
\begin{minipage}{.4\linewidth}
\texttt{iupac}{%
Tetra|cyclo[2.2.2.1^{1,4}]-un|decane-2-dodecyl-%
5-\{hepta|decyl|iso|dodecyl|thio\}ester%
}\end{verbatim}

Tetracyclo[2.2.2.1^{4}]-undecane-2- dodecyl-5-(heptadecylisododecyl- thioester)

The \texttt{iupac} command is more of a semantic command. In many cases you can achieve (nearly) the same thing by using - instead of \texttt{|}, and \texttt{textsuperscript} instead of ^ without \texttt{iupac}. There are some important differences, though:

- The character - inserts a small space before the hyphen and removes a small space after it. Also usually words with hyphens are only allowed to break at the hyphen. Inside \texttt{iupac} the hyphen will not prevent further hyphenation. The amount of inserted space can be customized.

- The character | not only prevents ligatures but also inserts a small space. The amount of inserted space can be customized.

- The characters ( and ) allow the word to be hyphenated and don’t prevent further hyphenation, likewise [ and ].

- The character ’ is printed as \texttt{chemprime}.

\footnotetext[4]{The idea and initial implementation is shamelessly borrowed from bpchem by Bjørn Pedersen.}
5. User Modules

- The character = is printed as \nonbreakinghyphen.

\huge\iupac{2,4-Dichlorpentan} \par
\iupac{2,4-Dichlorpentan}

\texttt{\chemprime}

Prints a prime character in superscript position. It is defined as \ensuremath{{}^\prime}.

\texttt{\nonbreakinghyphen}

Prints a hyphen which doesn’t allow a linebreak after it. It is defined as \mbox{-}\nobreak\hspace{0pt}.

The spaces inserted by - and | can be customized.

\texttt{nomenclature} » \texttt{hyphen-pre-space} = \{dimm\}

Set the space that is inserted before the hyphen set with -. Default: .01em

\texttt{nomenclature} » \texttt{hyphen-post-space} = \{dimm\}

Set the space that is inserted after the hyphen set with -. Default: -.03em

\texttt{nomenclature} » \texttt{break-space} = \{dimm\}

Set the space inserted by |. Default: .01em

The command \iupac serves another purpose, too, however. Regardless of the setting of the \iupac option (see below) all the commands presented in this section are always defined inside \iupac. Quite a number of the naming commands have very general names: \meta, \D, \E, \L, \R, \S, \trans and so forth. This means they either are predefined already (\L \L) or are easily defined by another package or class (the cool package defines both \D and \E, for example). In order to give you control which commands are defined in which way, there is the option \iupac:

\texttt{nomenclature} » \iupac = auto|restricted|strict

Take care of how IUPAC naming commands are defined. Default: auto

It has three modes:

- \iupac = {auto}: if the commands are not defined by any package or class you’re using they are available generally, otherwise only inside \iupac.

- \iupac = {restricted}: all naming commands are only defined inside \iupac. If the commands are defined by another package they have that meaning outside. They’re not defined outside otherwise.

- \iupac = {strict}: \texttt{CHEMMCROS} overwrites any other definition and makes the commands available throughout the document. Of course the commands can be redefined (but only in the document body). They will still be available inside \iupac then.

Table 1 on the following page demonstrates the different modes.

---

5. Please read section 5.3.2 on the next page before you consider using the one-letter commands
5. User Modules

<table>
<thead>
<tr>
<th>Table 1: Demonstration of \iupac's modes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto</td>
</tr>
<tr>
<td>\L</td>
</tr>
<tr>
<td>\iupac{\L}</td>
</tr>
<tr>
<td>\D</td>
</tr>
<tr>
<td>\iupac{\D}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2: iupac shortcuts for Greek letters.</th>
</tr>
</thead>
<tbody>
<tr>
<td>macro</td>
</tr>
<tr>
<td>letter</td>
</tr>
</tbody>
</table>

5.3.2. Macros Defined (Not Only) For Usage in \iupac

**One-letter Macros** For some of the macros explained in this section one-letter commands are defined – with a caveat in mind, though: they are not actively recommended. One-letter commands seldomly have meaningful names and often they’ve also been defined by other packages. This means they make collaboration more difficult than it needs to be and are a source for package conflicts. \texttt{CHEMMACROS} solves the latter problem by only providing them inside the argument of \iupac. The one exception \texttt{CHEMMACROS} makes is the command \p (for things like pH) which is and will remain an official command (see section 5.1 starting on page 8). For all other one-letter macros alternatives with more meaningful names exist.

**Greek Letters** Greek letters in compound names are typeset upright. Here are a few examples for the existing macros:

\texttt{\chemalpha} α
Upright lowercase alpha

\texttt{\chembeta} β
Upright lowercase alpha

\texttt{\chemgamma} γ
Upright lowercase alpha

\texttt{\chemdelta} δ
Upright lowercase alpha

There exist two commands for each of the twenty-four Greek letters: a lowercase and an uppercase version (\chemalpha and \texttt{\chemAlpha}). Those commands are actually provided by the \texttt{CHEMGREEK} package. For more details read section 6.4 starting on page 29 and also refer to \texttt{CHEMGREEK}’s documentation.

There are a number of one-letter commands that some people may find convenient to use which use above mentioned commands to print Greek letters inside \iupac. They’re listed in table 2.

\texttt{\iupac{5\chemalpha-androstan-3\chembeta-ol}} \par
\texttt{\iupac{\chemalpha-(tri|chloro|methyl)-\chemomega \par}
\texttt{\chem-chloro|poly(1,4-phenylene|methylene)}}
5. User Modules

\[5\alpha\text{-androstan-3\beta-ol}\]
\[\alpha\text{-}(\text{trichloromethyl})\omega\text{-chloropoly(1,4-phenilenemethylene)}\]

**Hetero Atoms and added Hydrogen**  Attachments to hetero atoms and added hydrogen atoms are indicated by italic letters \([\text{Co}h+08]\). **CHEMMACROS** defines a few macros for the most common ones.

\[\text{hydrogen} \ H\]
The italic H for hydrogen. An alias for this command is \H.

\[\text{oxygen} \ O\]
The italic O for oxygen. An alias for this command is \O.

\[\text{nitrogen} \ N\]
The italic N for nitrogen. An alias for this command is \N.

\[\text{sulfur} \ S\]
The italic S for sulfur. An alias for this command is \Sf.

\[\text{phosphorus} \ P\]
The italic P for phosphorus. An alias for this command is \P.

\[\text{iupac}\{\text{nitrogen-methyl|benz|amide}\}\]
\[\text{iupac}\{\text{3\text{H-pyrrole}}\}\]
\[\text{iupac}\{\text{oxygen-ethyl hexanethioate}\}\]

**Cahn-Ingold-Prelog**

\[\text{cip}\{(\text{conf})\}\]
Typeset Cahn-Ingol-Prelog descriptors, e.g.: \text{cip}\{(R, S) \ (R,S)\}. \text{(conf)} may be a csv list of entries.

\[\text{rectus} \ (R)\]
The rectus descriptor. An alias for this command is \R.

\[\text{sinister} \ (S)\]
The sinister descriptor. An alias for this command is \S.

  Both these commands and the entgegen/zusammen descriptors get a small additional amount of kerning after the closing parenthesis. This amount can be changed through the following option:

  \text{nomenclature} » \text{cip-kern} = \{(\text{dim})\}    \text{Default: .075em}

  Set the amount of kerning after the closing parenthesis.

  The entries typeset by and implemented with \text{cip} can be customized further:

  \text{cip-outer-format} = \{(\text{format})\}    \text{Default: \text{upshape}}

  The format of parentheses and commas typeset by \text{cip}.

\text{Introduced in version 5.8}
\text{(2017/04/17)}
5. User Modules

\texttt{cip-inner-format} = \{(\texttt{format})\}

The format of the entries in \texttt{cip}. This format works additive to the outer format.

\texttt{cip-number-format} = \{(\texttt{format})\}

The format of numbers in \texttt{cip}. This format works additive to the outer format and is applied to arabic figures only.

\textbf{Fischer}

\texttt{dexter} \ D

The dexter descriptor. An alias for this command is \texttt{D}.

\texttt{laevus} \ L

The laevus descriptor. An alias for this command is \texttt{L}.

\textit{cis/trans, zusammen/entgegen, syn/anti \& tert}

- \texttt{cis cis} \ trans \textit{trans}
- \texttt{fac fac} \ mer \textit{mer}
- \texttt{sin sin} \ ter \textit{ter}
- \texttt{zusammen (Z)} \ entgegen (E)
- \texttt{syn syn} \ anti \textit{anti}
- \texttt{tert tert}

An alias for \texttt{entgegen} is \texttt{E} and an alias for \texttt{zusammen} is \texttt{Z}.

\textbf{ortho/meta/para}

\texttt{ortho o} \ meta \textit{m} \ para \textit{p}

Although these commands are provided I like to cite \textit{The IUPAC Blue Book} [PPR04]:

The letters \texttt{o}, \texttt{m}, and \texttt{p} have been used in place of \texttt{ortho}, \texttt{meta}, and \texttt{para}, respectively, to designate the 1,2-, 1,3-, and 1,4- isomers of disubstituted benzene. This usage is strongly discouraged and is not used in preferred IUPAC names. \footnote{PPR04, p. 90}

\textbf{Absolute Configuration}

\texttt{\textbackslash{}Rconf\{(letter)\}}

\texttt{\textbackslash{}Rconf: \textcircled{R} \ Rconf[]: (\textcircled{R})}

\texttt{\textbackslash{}Sconf\{(letter)\}}

\texttt{\textbackslash{}Sconf: \textcircled{S} \ Sconf[]: (\textcircled{S})}
5. User Modules

**Coordination Chemistry**  **CHEMMACROS** provides a few commands useful in coordination chemistry:

\bridge{(num)} $\mu_3$-
Denote bridging ligand connection.

\hapto{(num)} $\eta^5$-
Denote hapticity.

\dento{(num)} $\kappa^2$-
Denote denticity.

\[ \text{Ferrocene} = \text{iupac} \{ \text{bis(} \text{\hapto{5} cyclo|penta|dienyl)} \text{iron} \} \par \]
\[ \text{iupac} \{ \text{tetra-\bridge{3}iodido-tetrakis[tri|methyl|platinum(IV)]} \} \]

Ferrocene = bis($\eta^5$-cyclopentadienyl)iron
tetra-$\mu_3$-iodido-tetrakis[trimethylplatinum(IV)]

Two options allow customization:

- **nomenclature.bridge-number** = sub|super
  Default: sub
  Appends the number as a subscript or superscript, depending on the choice. The IUPAC recommendation is the subscript [Con+05].

- **nomenclature.coord-use-hyphen** = true|false
  Default: true
  Append a hyphen to \hapto, \dento and \bridge or don’t.

The default behaviour of \hapto and \dento has changed with version 5.8 to follow IUPAC recommendations.

**Examples**

\[ \text{iupac}\{\text{dexter-Wein|s"aure} \} = \]
\[ \text{iupac}\{\text{cip(25,3S)-Wein|s"aure} \} \par \]
\[ \text{iupac}\{\text{dexter-}(-$-$)-Threose} = \]
\[ \text{iupac}\{\text{cip(25,3R)-}(-$-$)-2,3,4-Tri|hydroxy|butanal} \par \]
\[ \text{iupac}\{\text{cis-2-Butene} \} = \]
\[ \text{iupac}\{\text{zusammen-2-Butene}, \par \]
\[ \text{iupac}\{\text{cip(2E,4Z)-Hexa|dien}e} \par \]
\[ \text{iupac}\{\text{meta-Xylo}l \} = \]
\[ \text{iupac}\{1,3-}Di|methyl|benzene} \]

-D-Weinsäure = (2S,3S)-Weinsäure
D-(−)-Threose = (2S,3R)-(−)-2,3,4-Trihydroxybutanal
\text{cis}-2-Butene = (Z)-2-Butene,
(2E,4Z)-Hexadiene
m-Xylo = 1,3-Dimethylbenzene

5.3.3. Own \texttt{iupac} Macros And Shorthands

If you find any commands missing you can define them using
5. User Modules

\NewChemIUPAC{⟨cs⟩}{⟨declaration⟩}
Define a new \texttt{iupac} command that is in any case defined inside of \texttt{iupac} regardless if \langle cs \rangle is defined elsewhere already.

\ProvideChemIUPAC{⟨cs⟩}{⟨declaration⟩}
Define a new \texttt{iupac} command that is in any case defined inside of \texttt{iupac} regardless if \langle cs \rangle is defined elsewhere already only if the corresponding \texttt{iupac} macro is not defined, yet.

\RenewChemIUPAC{⟨cs⟩}{⟨declaration⟩}
Redefine an existing \texttt{iupac} command that is in any case defined inside of \texttt{iupac} regardless if \langle cs \rangle is defined elsewhere already.

\DeclareChemIUPAC{⟨cs⟩}{⟨declaration⟩}
Define a new \texttt{iupac} command that is in any case defined inside of \texttt{iupac} regardless if \langle cs \rangle is defined elsewhere already. This silently overwrites an existing \texttt{iupac} macro definition.

\LetChemIUPAC{⟨cs1⟩}{⟨cs2⟩}
Defines \langle cs1⟩ to be an alias of \langle cs2⟩.

A command defined in this way will obey the setting of the option \texttt{iupac}. This means any existing command is only overwritten with \texttt{iupac = {strict}}. However, \texttt{\NewChemIUPAC} will not change the definition of an existing \texttt{iupac} naming command but issue an error if the \texttt{iupac} naming command already exists. \texttt{\DeclareChemIUPAC} will overwrite an existing \texttt{iupac} command.

\begin{quote}
\begin{verbatim}
\NewChemIUPAC\textsc{endo}\{textsc{endo}\}
\RenewChemIUPAC\textsc{anti}\{textsc{anti}\}
\texttt{iupac}{(2-ENDO,7-ANTI)-2-bromo-7-fluorobicyclo[2.2.1]heptane}
\end{verbatim}
\end{quote}

\RenewChemIUPAC allows you to redefine the existing \texttt{iupac} naming commands.

\begin{quote}
\begin{verbatim}
\iupac{\texttt{meta-Xylol}} \par
\RenewChemIUPAC\textsc{meta}\{\texttt{textup{m}}\}
\iupac{\texttt{meta-Xylol}}
\end{verbatim}
\end{quote}

There’s also a way for defining new \texttt{iupac} shorthands or changing the existing ones:

\NewChemIUPACShorthand{⟨shorthand token⟩}{⟨control sequence⟩}
Defines a new \texttt{iupac} shorthand. Inside \texttt{iupac} it will be equal to using \langle control sequence \rangle. This throws an error if \langle shorthand token \rangle is already defined.

\RenewChemIUPACShorthand{⟨shorthand token⟩}{⟨control sequence⟩}
Redefines an existing \texttt{iupac} shorthand. This throws an error if \langle shorthand token \rangle is not defined, yet.

\DeclareChemIUPACShorthand{⟨shorthand token⟩}{⟨control sequence⟩}
Defines a new \texttt{iupac} shorthand or redefines an existing one.

\ProvideChemIUPACShorthand{⟨shorthand token⟩}{⟨control sequence⟩}
Provides a new \texttt{iupac} shorthand. Does nothing if \langle shorthand token \rangle is already defined.
5. User Modules

\RemoveChemIUPACShorthand{shorthand token}
Deletes an existing IUPAC shorthand.

5.3.4. Latin Phrases

CHEMMACROS provides a command for typesetting latin phrases:

\latin\{\{options\}\}\{\{phrase\}\}
Typesets \{phrase\} according to the option format described below.

\insitu \textit{in situ}

\invacuo \textit{in vacuo}

\abinitio \textit{ab initio}

If you additionally load chemstyle [Wri13] said package will not define its own \latin. The last three commands mentioned above are defined through

\NewChemLatin\{\{cs\}\}\{\{phrase\}\}
Define a new latin phrase. Gives an error if \{cs\} already exists.

\DeclareChemLatin\{\{cs\}\}\{\{phrase\}\}
Define a new latin phrase. Silently redefined existing macros.

\RenewChemLatin\{\{cs\}\}\{\{phrase\}\}
Redefine an existing latin phrase. Gives an error if \{cs\} doesn’t exist.

\ProvideChemLatin\{\{cs\}\}\{\{phrase\}\}
Define a new latin phrase only if \{cs\} doesn’t exist.

\begin{code}
\NewChemLatin\ltn{latin text}\ltn{\textit{latin text}}
\end{code}

You can change the appearance with this option:

\texttt{nomenclature = format = \{\{definition\}\}} \hspace{2cm} \texttt{Default: \textit{emph}}
Sets the format for the latin phrases.

5.4. The particles Module

The particles module loads the modules charges and chemformula.

5.4.1. Provided Particle Macros

The particles defines a number of macros which can be used for typesetting common particles in the running text. Most of them don’t make much sense in chemformula [Nie19]’s \ch, though, which doesn’t mean that they can’t be used there, of course:

\begin{verbatim}
\el e^- \prt p^+ \ntr n^0 \Hyd OH^- \Oxo H_2O^+ \water H_2O \El E^- \Nuc Nu^- \ba ba^-
\end{verbatim}
All of these macros are defined using chemformula’s \chcpd. The details are explained in section 5.4.2.

The macros \Nuc and \ba are special: they have an optional argument for the following options:

- **elpair** = dots|dash|false  
  Default: false
  Determine how the electron pair of the nucleophiles is displayed. The electron pair is drawn using CHEMFORMULA’s \chlewis macro.

- **space** = {...dim}  
  Default: .1em
  Sets the space that is inserted between the electron pair and the negative charge sign.

Both options can of course also be set with \chemsetup.

---

5.4.2. Defining Own Particle Macros

There are two sets of macros, one for defining particles and one for defining nucleophiles.

- \NewChemParticle{(cs)}{(formula)}  
  Defines a new macro (cs). (formula) is any valid CHEMFORMULA input (this depends on the setting of the formula option, see 6.2 starting on page 27). Raises an error if (cs) already exists.

- \RenewChemParticle{(cs)}{(formula)}  
  Redefines a new macro (cs). (formula) is any valid CHEMFORMULA input (this depends on the setting of the formula option, see 6.2 starting on page 27). Raises an error if (cs) doesn’t exist.

- \DeclareChemParticle{(cs)}{(formula)}  
  Defines a macro (cs). (formula) is any valid CHEMFORMULA input (this depends on the setting of the formula option, see 6.2 starting on page 27). Silently overwrites (cs) if it exists.

- \ProvideChemParticle{(cs)}{(formula)}  
  Defines a new macro (cs). (formula) is any valid CHEMFORMULA input (this depends on the setting of the formula option, see 6.2 starting on page 27). Does nothing if (cs) already exists.

An example of usage is the definition of the existing particle macros:

\begin{verbatim}
\NewChemParticle\el {e-} \Nuc[elpair=dash] \Nuc[elpair=dash]
\ba\Nuc\ba\Nuc
\end{verbatim}

The following set defines macros like \Nuc

\begin{verbatim}
\NewChemNucleophile{(cs)}{(formula)}
\end{verbatim}

Defines a new macro (cs). (formula) is any valid CHEMFORMULA input (this depends on the setting of the formula option, see 6.2 starting on page 27). Note that (formula) will get a trailing negative charge! Raises an error if (cs) already exists.
Redefines a new macro \(\text{(cs)}\). \(\text{(formula)}\) is any valid \text{CHEMFORMULA} (this depends on the setting of the \text{formula} option, see 6.2 starting on page 27). Note that \(\text{(formula)}\) will get a trailing negative charge! Raises an error if \(\text{(cs)}\) doesn’t exist.

\text{\texttt{\textbackslash DeclareChemNucleophile\{}{(cs)}\texttt{\{}{(formula)}\texttt{}}}\}

Defines a macro \(\text{(cs)}\). \(\text{(formula)}\) is any valid \text{CHEMFORMULA} (this depends on the setting of the \text{formula} option, see 6.2 starting on page 27). Note that \(\text{(formula)}\) will get a trailing negative charge! Silently overwrites \(\text{(cs)}\) if it exists.

\text{\texttt{\textbackslash ProvideChemNucleophile\{}{(cs)}\texttt{\{}{(formula)}\texttt{}}}\}

Defines a new macro \(\text{(cs)}\). \(\text{(formula)}\) is any valid \text{CHEMFORMULA} (this depends on the setting of the \text{formula} option, see 6.2 starting on page 27). Note that \(\text{(formula)}\) will get a trailing negative charge! Does nothing if \(\text{(cs)}\) already exists.

An example of usage is the definition of the existing nucleophile macros:

\text{\begin{verbatim}
\NewChemNucleophile{\text{Nu}}{\text{Nu}}
\NewChemNucleophile{\text{ba}}{\text{ba}}
\end{verbatim}}

A macro defined this way will have an optional argument for the \text{elpair} option.

5.5. The phases Module

The phases module loads the \text{chemformula} modul.

5.5.1. Basics

These commands are intended to indicate the phase of a compound.

\text{\texttt{\textbackslash sld\{}{(s)}\texttt{\{}\texttt{\textbackslash lqd\{}{(l)}\texttt{\{}\texttt{\textbackslash gas\{}{(g)}\texttt{\{}\texttt{\textbackslash aq\{}{(aq)}}\texttt{}}}\}

\text{\begin{verbatim}
\ch{C\texttt{\textbackslash sld\{}{\texttt{\textbackslash lqd\{}{2\texttt{\textbackslash lqd\{}H2O\texttt{\textbackslash lqd\{}\texttt{\textbackslash ->\texttt{\textbackslash CO2\texttt{\textbackslash gas\}}\texttt{\{}{\texttt{\textbackslash gas\{}{2\texttt{\textbackslash gas\{}H2\texttt{\textbackslash gas}}\texttt{\}}}\texttt{\par}}
To make it complete: NaCl\texttt{\textbackslash aq}.
C\texttt{\textbackslash sld\{}{l}\texttt{\textbackslash lqd\{}{2H2O\texttt{\textbackslash lqd\{}\texttt{\textbackslash ->\texttt{\textbackslash CO2\texttt{\textbackslash gas\}}\texttt{\{}{\texttt{\textbackslash gas\{}{2H2\texttt{\textbackslash gas}}\texttt{\}}}\texttt{\par}}
To make it complete: NaCl\texttt{\textbackslash aq}.
\end{verbatim}}

The IUPAC recommendation to indicate the state of aggregation is to put it in parentheses after the compound [Coh+08]. However, you might want to put it as a subscript which is also very common.

The [...] symbols are used to represent the states of aggregation of chemical species. The letters are appended to the formula in parentheses and should be printed in Roman (upright) type without a full stop (period). The IUPAC Green Book [Coh+08, p. 54]

There are two options to customize the output:

\text{\texttt{pos = side|sub \hspace{1cm} phases \texttt{\{}\texttt{}}} \hspace{1cm} Default: side}

Switch the position of the phase indicator.

\text{\texttt{space = {{\texttt{\textbackslash dim}}}} \hspace{1cm} phases \texttt{\{}\texttt{}}} \hspace{1cm} Default: .1333em}

Change the default spacing between compound a phase indicator if \texttt{pos = \{} \texttt{side}. A \TeX{} dimension.
To make it complete: NaCl\aq.

\chem{C(s) + 2H_2O(l) \rightarrow CO_2(g) + 2H_2(g)}

To make it complete: NaCl\aq.

All those phase commands have an optional argument:

\chem{H_2O \ "lqd\[SI\{5\\}\celsius\]} H_2O(l, 5 °C)

There is also a generic phase command:

\phase{\(phase\)}

If you need a phase indicator just once or twice. You can use it to denote a phase for which there is no phase command, yet.

5.5.2. Define Own Phases

Depending on the subject of your document you might need to indicate other states of aggregation. You can easily define them.

\NewChemPhase{\(cs\)}{\(symbol\)}

Define a new phase command. See section 5.5.3 on the next page for a way to define language dependent settings. Gives an error if \(cs\) already exists.

\DeclareChemPhase{\(cs\)}{\(symbol\)}

Define a new phase command. See section 5.5.3 on the following page for a way to define language dependent settings. Overwrites previous definitions of \(cs\).

\RenewChemPhase{\(cs\)}{\(symbol\)}

Redefine an existing phase command. See section 5.5.3 on the next page for a way to define language dependent settings. Gives an error if \(cs\) is not defined.

\ProvideChemPhase{\(cs\)}{\(symbol\)}

Define a new phase command. See section 5.5.3 on the following page for a way to define language dependent settings. Does nothing if \(cs\) is already defined.

\% preamble:
\NewChemPhase{aq}{\infty} % aqueous solution at infinite dilution
\NewChemPhase{cd}{\text{cd}} % condensed phase
\NewChemPhase{lc}{\text{lc}} % liquid crystal
\chem{NaOH\aq} \chem{H_2O\cd} \chem{U\phase{cr}} \chem{A\lc}
\chemsetup[phases]{pos=sub}
\chem{NaOH\aq} \chem{H_2O\cd} U\text{(cr)} A\text{(lc)}
NaOH\aq \H_2\O\cd U\text{(cr)} A\text{(lc)}
5.5.3. Language Dependencies

For each phase command a translation into the custom language can be defined. If a phase is declared with `\NewChemPhase` no translation exists and for every babel language the literal string is used that was provided as a definition. Let’s say you define the phase

\begin{verbatim}
\NewChemPhase{liquid}{l}
\end{verbatim}

and want to add the German translation “f{l}”. Then you could do

\begin{verbatim}
\DeclareTranslation{German}{phase-liquid}{f{l}}
\end{verbatim}

This way, when you use it in a German document using the appropriate babel option using `\liquid` would correctly translate. For this the package translations [Nie20] is used. The ID always is `phase-⟨csname⟩` where ⟨csname⟩ is the name of the phase command you defined without leading backslash.

See section 6.5 starting on page 30 for predefined translations and general language options of [CHEMMACROS].

5.6. The symbols Module

The symbols module defines a few symbols chemists need now and then. It loads the package amstext [MSo0].

- `\transitionstatesymbol` This is self-explaining: ♣
- `\standardstate` Again self-explaining: ♣
- `\changestate` The uppercase delta used in ΔH for example.

6. Internal Modules

6.1. The base Module

The base module is the core module of [CHEMMACROS]. It defines some tools which can (and should) be used in other modules. This means this section is only interesting for you if you plan to write a module yourself (see section A starting on page 65 for details).

This module requires the packages bm [CM19], amstext [MSo0], and etoolbox [Leh19]. This module also provides `\chemsetup` and the option `modules`.

It also provides a number of (expl3) macros which may be used in other modules. In the macro descriptions below `TF` denotes that a T, an F and a TF variant exist. In case of an expandable conditional (∗) also the predicate variant is available.

- `\chemmacros_if_loaded:nnTF {package|class} ⟨name⟩ ⟨true⟩ ⟨false⟩` Checks if package (or class) ⟨name⟩ has been loaded. Also works after begin document.
- `\chemmacros_if_package_loaded:nTF ⟨name⟩ ⟨true⟩ ⟨false⟩` Checks if package ⟨name⟩ has been loaded. Also works after begin document.
6. Internal Modules

\chemmacros_if_class_loaded:nTF \{\textit{name}\} \{\textit{true}\} \{\textit{false}\}
Checks if class \textit{name} has been loaded. Also works after begin document.

\chemmacros_nobreak:
Inserts a penalty of 10 000.

\chemmacros_allow_break:
Inserts a penalty of 0.

\chemmacros_skip_nobreak:N \{\text{skip/length variable}\}
Insert a horizontal skip where a linebreak is disallowed.

\chemmacros_if_is_int:nTF \{\textit{input}\} \{\textit{true}\} \{\textit{false}\}
Checks if \textit{input} is an integer or something else.

\chemmacros_if_bold:TF \{\textit{true}\} \{\textit{false}\}
Checks if the current font weight is one of b, bc, bm, bx, eb, ebc, mb, sb, sbc, sbx, ub, ubc or ubx.

\chemmacros_bold:n \{\textit{text}\}
Checks if the current font weight is bold and if yes places \textit{text} in \textbf{if in text mode or in } \bm{if in math mode. If no } \textit{text} simply is placed in the input stream as is.

\chemmacros_text:n \{\textit{text}\}
Ensures that \textit{text} is placed in text mode.

\chemmacros_math:n \{\textit{text}\}
Ensures that \textit{text} is placed in math mode.

\chemmacros_new_macroset:nnn \{\textit{name}\} \{\textit{arg spec}\} \{\textit{internal command call}\}
A command to define a set of macros \NewChem\textit{name}, \RenewChem\textit{name}, \DeclareChem\textit{name} and \ProvideChem\textit{name} where the first letter of \textit{name} is converted to uppercase, other letters are kept unchanged. \textit{arg spec} is any valid argument specification for xparse’s \DeclareDocumentCommand [L3Pb]. \textit{internal command call} should be a macro which makes definitions without error checks, i.e., define new macros or redefine existing ones like \def does. This macro just should get the arguments passed on to. Have a look at the example below.

\chemmacros_new_environment_macroset:nnn \{\textit{name}\} \{\textit{arg spec}\} \{\textit{internal command call}\}
Like \chemmacros_new_macroset:nnn but for environments.

\NewChemMacroset*\{\textit{name}\} \{\textit{arg spec}\} \{\textit{internal command call}\}
A non-expl3 version of \chemmacros_new_macroset:nnn for \LaTeX 2e programmers. The starred version calls \chemmacros_new_environment_macroset:nnn.

\chemmacros_add_cleveref_support:nnnnn \{\textit{counter}\} \{\textit{singular}\} \{\textit{plural}\} \{\textit{uppercase singular}\} \{\textit{uppercase plural}\}
A command to add suiting names for a counter for the cleveref package’s \texttt{cref} commands. This command acts at the end of the preamble and only if a user hasn’t provided definitions with \texttt{crefname} already.

\ChemCleverefSupport \{\textit{counter}\} \{\textit{singular}\} \{\textit{uppercase singular}\} \{\textit{plural}\} \{\textit{uppercase plural}\}
\LaTeX 2e-version of \chemmacros_add_cleveref_support:nnnnn.
6. Internal Modules

\chemmacros_add_fancyref_support:nnn \{\textit{prefix}\} \{\textit{name}\} \{\textit{uppercase name}\}
A command to add suitting names for a counter for the fancyref package’s \fref commands. This command acts at the end of the preamble and doesn’t override definitions made by the users.

\ChemFancyrefSupport\{\textit{prefix}\}\{\textit{name}\}\{\textit{uppercase name}\}
\LaTeX\ 2e\text{-}version of \chemmacros_add_fancyref_support:nnnnn.

This is how the macros \NewChemParticle, \RenewChemParticle, \DeclareChemParticle and \ProvideChemParticle were defined:

\begin{verbatim}
\NewChemMacroset \{\textit{Particle}\} \{\textit{mm}\}
{ \chemmacros_define_particle:Nn \textit{#1} \{\textit{#2}\} }
\end{verbatim}

The following macros strictly speaking are not provided by the base module but this place fits best for their description.

* \chemmacros_if_module_exist:nTF \{\textit{module}\} \{\textit{true}\} \{\textit{false}\}
Checks if a file with the correct name for a module \textit{module} can be found.

* \chemmacros_if_module_loaded:nTF \{\textit{module}\} \{\textit{true}\} \{\textit{false}\}
Checks if the module \textit{module} has already been loaded or not.

\chemmacros_load_module:n \{\textit{module}\}
Loads module \textit{module} if it hasn’t been loaded, yet.

\chemmacros_load_modules:n \{\textit{csv list of modules}\}
Loads every module in \textit{csv list of modules} if they haven’t been loaded, yet. This is the code level variant of \usechemmodule.

\chemmacros_before_module:nn \{\textit{module}\} \{\textit{code}\}
Saves \textit{code} and inserts it right before \textit{module} is loaded. If \textit{module} is never loaded then \textit{code} is never inserted. If \textit{module} already is loaded when the command is used then \textit{code} also is never inserted.

\chemmacros_after_module:nn \{\textit{module}\} \{\textit{code}\}
Saves \textit{code} and inserts it right after \textit{module} is loaded. If \textit{module} is never loaded then \textit{code} is never inserted. If \textit{module} already is loaded when the command is used then \textit{code} is inserted immediately.

6.2. The \texttt{chemformula} Module
The \texttt{chemformula} module loads the amstext package [MSoo] and the \texttt{charges} module.

6.2.1. For Users
There are different packages which provide means for typesetting chemical formulas:

* \texttt{chemformula} [Nie19]. This is probably well known to users of \texttt{CHEMMACROS}.

* \texttt{mhchem} [Hen18]. This is the “older brother” of \texttt{CHEMFORMULA}.

* \texttt{chemfig} [Tel19]. The easiest and most complete of the packages for drawing skeletal formulas.
• \textsc{X\textregistered\textsc{M}TEX [Fuj13].} A very comprehensive alternative for typesetting chemistry.

In order to help authors getting a consistent layout \textsc{chemmacros} does not make a choice which package to use for typesetting formulas. Although \textsc{chemformula} is well tested and preferred users can choose other packages if they like.

This is done with the following general option:

\begin{verbatim}
formula = {⟨method⟩}
\end{verbatim}

Default: \textsc{chemformula}

This option let’s you choose how chemical formulas are typeset. Available methods are

- \textsc{chemformula}
- \textsc{mhchem}
- \textsc{chemist} (from the \textsc{X\textregistered\textsc{M}TEX bundle})
- \textsc{chemfig}

The corresponding package with the same name is loaded.

If you explicitly set this option the corresponding package is loaded immediately and the method is set up. Otherwise the option will be set by \textsc{chemmacros} at the end of the preamble.

If you load a method package in a way that a unique choice is possible then \textsc{chemmacros} will set the method accordingly if you haven’t set the option by yourself. If no unique choice is possible \textsc{chemmacros} will raise a warning and choose \textsc{chemformula} regardless if the package is loaded or not. In this case if you want to use another method you’ll have to choose manually. \textit{All automatic choices only happen at the end of the preamble.}

\textbf{Using the chemformula Package} If you set formula = \{chemformula\} the \textsc{chemformula} module makes it possible that you can set all \textsc{chemformula} options via the \texttt{\chemsetup} command using the module \textsc{chemformula}, for example:

\begin{verbatim}
\chemsetup[chemformula]{format=\sffamily}
\end{verbatim}

Everywhere where \textsc{chemmacros} typesets chemical formulas \textsc{chemformula}’s macros \texttt{\chcpd} or \texttt{\ch} are used, for example in the reaction environments provided by the \texttt{reactions} module.

\textit{This method is the recommended choice!}

\textbf{Using the mhchem Package} If you set formula = \{mhchem\} the \textsc{chemformula} module makes it possible that you can set all of \textsc{mhchem}’s options via the \texttt{\chemsetup} command using the module \textsc{mhchem}, for example:

\begin{verbatim}
\chemsetup[mhchem]{format=\sffamily}
\end{verbatim}

Everywhere where \textsc{chemmacros} typesets chemical formulas \textsc{mhchem}’s macro \texttt{\ce} is used, for example in the reaction environments provided by the \texttt{reactions} module.

There are some \textit{caveats} if you use this method:

- This method has not been extensively tested, yet. There may be errors and wrong output at unexpected places.
6. Internal Modules

- Using this method effectively disables the different values of the \texttt{particles} option \texttt{elpair} (see section 5.4).

- The different kinds of formal charges provided by the \texttt{charges} module (see section 5.2.2) are disabled. Formal charges always use the math method now.

- There may also be other incompatibilities (e.g., \texttt{mhchem} has its own method of setting upright Greek letters so it may or may not disable \texttt{CHEMMACROS}' mechanism).

**Using the chemfig Package**  
Everywhere where \texttt{CHEMMACROS} typesets chemical formulas chemfig's macro \texttt{\printatom} is used, for example in the reaction environments provided by the \texttt{reactions} module.

There are some **caveats** if you use this method:

- This method has not been extensively tested, yet. There may be errors and wrong output at unexpected places.

- Using this method effectively disables the different values of the \texttt{particles} option \texttt{elpair} (see section 5.4).

- The different kinds of formal charges provided by the \texttt{charges} module (see section 5.2.2) are disabled. Formal charges always use the math method now.

- The reaction environments by the \texttt{reactions} module may work only to a limited respect. If you plan to use them consider using methods \texttt{chemformula} or \texttt{mhchem} instead.

**Using the chemist Package**  
Everywhere where \texttt{CHEMMACROS} typesets chemical formulas chemist's macro \texttt{\ChemForm} is used, for example in the reaction environments provided by the \texttt{reactions} module.

There are some **caveats** if you use this method:

- This method has not been extensively tested, yet. There may be errors and wrong output at unexpected places.

- Using this method effectively disables the different values of the \texttt{particles} option \texttt{elpair} (see section 5.4).

- The different kinds of formal charges provided by the \texttt{charges} module (see section 5.2.2) are disabled. Formal charges always use the math method now.

- The reaction environments by the \texttt{reactions} module may work only to a limited respect. If you plan to use them consider using methods \texttt{chemformula} or \texttt{mhchem} instead.

### 6.2.2. For Module Writers

There are two macros for module writers:

\texttt{\chemmacros\_chemformula:n \{(formula)\}}

This is only a wrapper for \texttt{\chcpd} or \texttt{\ce}. It is recommended that module writers use this macro (or a variant thereof) inside of \texttt{CHEMMACROS'} macros whenever they want to display a chemical formula. Writers who prefer traditional \LaTeX programming over expl3 should use \texttt{\chemmacros@formula}.

---

6. On the other hand \TeX (and especially the chemist package) provides quite a number of chemical reaction environments itself.
6. Internal Modules

\chemmacros_reaction: \{(reaction)\}
This is only a wrapper for \ch or \ce. It is recommended that module writers use this macro (or a variant thereof) inside of \textsc{chemmacros}' macros whenever they want to display a chemical reaction. Writers who prefer traditional \LaTeX programming over expl3 should use \textsc{chemmacros@reaction}.

6.3. The \texttt{errorcheck} Module

The \texttt{errorcheck} module provides some rudimentary support for giving users more meaningful messages when they use a command or environment provided by a module that they haven’t loaded.

6.4. The \texttt{greek} Module

The \texttt{greek} module loads the \texttt{chemgreek} package [Nie16a].

This module provides one option:

\texttt{greek = \{(mapping)\}}
A valid value is any valid \texttt{chemgreek (mapping)}. \textsc{chemmacros} will warn you if no mapping has been chosen or if you are using the \texttt{default} or the \texttt{var-default} mapping because this means that no upright Greek letters are available.

If you load a \texttt{chemgreek} support package which allows an unambiguous choice of a mapping \texttt{chemgreek} will make this choice automatically. This means if you say

\begin{verbatim}
1. \usepackage{upgreek}
2. \usepackage{chemmacros}
\end{verbatim}
then \textsc{chemmacros} will use upgreek’s upright Greek letters. If you have

\begin{verbatim}
1. \usepackage{upgreek}
2. \usepackage{chemmacros}
3. \usepackage{textgreek}
\end{verbatim}
then no unambiguous choice is possible and you should choose a mapping yourself, for example:

\begin{verbatim}
1. \usepackage{upgreek}
2. \usepackage{chemmacros}
3. \usepackage{textgreek}
4. \chemsetup{greek=textgreek}
\end{verbatim}
For further details on mappings please refer to \texttt{chemgreek}’s manual.

6.5. The \texttt{lang} Module

The \texttt{lang} module provides language support for \textsc{chemmacros}. It loads the package translations [Nie20].
6. Internal Modules

6.5.1. Information For Users

This module defines the following option:

\texttt{language = \texttt{auto}}\mid \texttt{\textlangle language\textrangle}

\begin{itemize}
  \item \texttt{language = \texttt{auto}}
  \begin{itemize}
    \item \texttt{\texttt{language}} will detect the language used by\texttt{ babel} [\texttt{Bra19}] or\texttt{ polyglos-}
    \item \texttt{sia} [\texttt{Cha19}] automatically, the fallback translation is\texttt{ English} and will be\texttt{ used}
    \item \texttt{if no translation for the actual language is available. Any language known to the translation}
    \item \texttt{package is a valid value for \textlangle language\textrangle}.\end{itemize}
\end{itemize}

The language chosen via \texttt{language} is used for\texttt{ translation} of certain strings in different places all\texttt{ over chemmacros}. They are mentioned in the places when the corresponding function of\texttt{ chemmacros} is explained.

Translation is done with the help of the translations package, available translation keys are\texttt{ listed} in section 6.5.2.

6.5.2. Available Translation Keys

Table 3 on the following page lists all predefined translations of the available keys. \textit{Some of the \texttt{translations} have changed in version 5.6}. The \texttt{lang} module doesn’t provide the translations themselves – they are provided by the corresponding modules. A translation key is a unique\texttt{ string} of characters. Each key is used to identify a replacement text which depends on the current language or the language set through the \texttt{language} option. For each key at least the\texttt{ English} fallback translation is provided, for most also the German translation is provided. For a few keys also other translations are provided. If you find that a translation for your language is missing you can provide it in the preamble:

\begin{itemize}
  \item \texttt{\textbackslash DeclareChemTranslation}{\{key\}}{\{language\}}{\{translation\}}
  \begin{itemize}
    \item \texttt{\textbackslash DeclareChemTranslation} is a command which makes an abstraction from the translations package. It should be used in documents for adding missing translations that are needed. This command can only be used in the preamble.
  \end{itemize}
\end{itemize}

\begin{itemize}
  \item \texttt{\textbackslash DeclareChemTranslations}{\{key\}}{\{language\} = \{translation\}}
  \begin{itemize}
    \item \texttt{\textbackslash DeclareChemTranslations} is a command rather meant for module writers but can be used by document authors as well, of course. It gets a csv list of key/value pairs of translations. This command can only be used in the preamble.
  \end{itemize}
\end{itemize}

If you send me an email (see section B starting on page 66) with the translations for your language I’ll gladly add them to the next release of\texttt{ chemmacros}!

\textit{Please do not use translations’ \texttt{\textbackslash DeclareTranslation} for declaring translations.}

\begin{table}[h]
\centering
\begin{tabular}{llll}
\hline
key & language & translation \\
\hline
K-acid & fallback & \texttt{\textbackslash mathrm \{a\}} \\
K-base & fallback & \texttt{\textbackslash mathrm \{b\}} \\
K-water & fallback & \texttt{\textbackslash mathrm \{w\}} \\
K-acid & German & \texttt{\textbackslash mathrm \{s\}} \\
\hline
\end{tabular}
\caption{Translation keys predefined by\texttt{ chemmacros} (except phase-aqi, phase-cd and phase-lc which were defined in this document).}
\end{table}

7. That is, a string using the definition for strings used for expl3, i.e., converted to a series of category code 12 characters.
### Key Language Translation

<table>
<thead>
<tr>
<th>Key</th>
<th>Language</th>
<th>Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-acid</td>
<td>Dutch</td>
<td>$\mathbb{z}$</td>
</tr>
<tr>
<td>phase-sld</td>
<td>fallback</td>
<td>s</td>
</tr>
<tr>
<td>phase-lqd</td>
<td>fallback</td>
<td>l</td>
</tr>
<tr>
<td>phase-gas</td>
<td>fallback</td>
<td>g</td>
</tr>
<tr>
<td>phase-aq</td>
<td>fallback</td>
<td>aq</td>
</tr>
<tr>
<td>phase-sld</td>
<td>German</td>
<td>f</td>
</tr>
<tr>
<td>phase-lqd</td>
<td>German</td>
<td>f/l</td>
</tr>
<tr>
<td>list-of-reactions</td>
<td>fallback</td>
<td>List of Reactions</td>
</tr>
<tr>
<td>list-of-reactions</td>
<td>English</td>
<td>List of Reactions</td>
</tr>
<tr>
<td>list-of-reactions</td>
<td>German</td>
<td>Reaktionsverzeichnis</td>
</tr>
<tr>
<td>list-of-reactions</td>
<td>Italian</td>
<td>Elenco delle reazioni</td>
</tr>
<tr>
<td>list-of-reactions</td>
<td>French</td>
<td>Table des r'{e}actions</td>
</tr>
<tr>
<td>list-of-reactions</td>
<td>Dutch</td>
<td>Lijst van reacties</td>
</tr>
<tr>
<td>list-of-reactions</td>
<td>Norwegian</td>
<td>Reaksjonsliste</td>
</tr>
<tr>
<td>list-of-reactions</td>
<td>Nynorsk</td>
<td>Reaksjonsliste</td>
</tr>
<tr>
<td>reaction</td>
<td>fallback</td>
<td>reaction</td>
</tr>
<tr>
<td>reaction</td>
<td>English</td>
<td>reaction</td>
</tr>
<tr>
<td>reaction</td>
<td>German</td>
<td>Reaktion</td>
</tr>
<tr>
<td>reaction</td>
<td>Italian</td>
<td>reazione</td>
</tr>
<tr>
<td>reaction</td>
<td>French</td>
<td>r'{e}actions</td>
</tr>
<tr>
<td>reaction</td>
<td>Dutch</td>
<td>reactie</td>
</tr>
<tr>
<td>reaction</td>
<td>Norwegian</td>
<td>reaksjon</td>
</tr>
<tr>
<td>reaction</td>
<td>Nynorsk</td>
<td>reaksjon</td>
</tr>
<tr>
<td>reactions</td>
<td>fallback</td>
<td>reactions</td>
</tr>
<tr>
<td>reactions</td>
<td>English</td>
<td>reactions</td>
</tr>
<tr>
<td>reactions</td>
<td>German</td>
<td>Reaktionen</td>
</tr>
<tr>
<td>reactions</td>
<td>Italian</td>
<td>reazioni</td>
</tr>
<tr>
<td>reactions</td>
<td>French</td>
<td>r'{e}actions</td>
</tr>
<tr>
<td>reactions</td>
<td>Dutch</td>
<td>reacties</td>
</tr>
<tr>
<td>reactions</td>
<td>Norwegian</td>
<td>reaksjoner</td>
</tr>
<tr>
<td>reactions</td>
<td>Nynorsk</td>
<td>reaksjoner</td>
</tr>
<tr>
<td>Reaction</td>
<td>fallback</td>
<td>Reaction</td>
</tr>
<tr>
<td>Reaction</td>
<td>English</td>
<td>Reaction</td>
</tr>
<tr>
<td>Reaction</td>
<td>German</td>
<td>Reaktion</td>
</tr>
<tr>
<td>Reaction</td>
<td>Italian</td>
<td>Reazione</td>
</tr>
<tr>
<td>Reaction</td>
<td>French</td>
<td>R'{e}actions</td>
</tr>
<tr>
<td>Reaction</td>
<td>Dutch</td>
<td>Reactie</td>
</tr>
<tr>
<td>Reaction</td>
<td>Norwegian</td>
<td>Reaksjon</td>
</tr>
<tr>
<td>Reaction</td>
<td>Nynorsk</td>
<td>Reaksjon</td>
</tr>
<tr>
<td>Reactions</td>
<td>fallback</td>
<td>Reactions</td>
</tr>
<tr>
<td>Reactions</td>
<td>English</td>
<td>Reactions</td>
</tr>
<tr>
<td>Reactions</td>
<td>German</td>
<td>Reaktionen</td>
</tr>
<tr>
<td>Reactions</td>
<td>Italian</td>
<td>Reazioni</td>
</tr>
<tr>
<td>Reactions</td>
<td>French</td>
<td>R'{e}actions</td>
</tr>
<tr>
<td>Reactions</td>
<td>Dutch</td>
<td>Reacties</td>
</tr>
<tr>
<td>Reactions</td>
<td>Norwegian</td>
<td>Reaksjoner</td>
</tr>
<tr>
<td>Reactions</td>
<td>Nynorsk</td>
<td>Reaksjoner</td>
</tr>
</tbody>
</table>

*continues*
6. Internal Modules

<table>
<thead>
<tr>
<th>key</th>
<th>language</th>
<th>translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheme-name</td>
<td>fallback</td>
<td>Scheme</td>
</tr>
<tr>
<td>scheme-name</td>
<td>English</td>
<td>Scheme</td>
</tr>
<tr>
<td>scheme-name</td>
<td>German</td>
<td>Schema</td>
</tr>
<tr>
<td>scheme-name</td>
<td>Norwegian</td>
<td>Skjema</td>
</tr>
<tr>
<td>scheme-name</td>
<td>Nynorsk</td>
<td>Skjema</td>
</tr>
<tr>
<td>scheme-list</td>
<td>fallback</td>
<td>List of Schemes</td>
</tr>
<tr>
<td>scheme-list</td>
<td>English</td>
<td>List of Schemes</td>
</tr>
<tr>
<td>scheme-list</td>
<td>German</td>
<td>Verzeichnis der Schemata</td>
</tr>
<tr>
<td>scheme-list</td>
<td>Norwegian</td>
<td>Skjemaliste</td>
</tr>
<tr>
<td>scheme-list</td>
<td>Nynorsk</td>
<td>Skjemaliste</td>
</tr>
<tr>
<td>scheme</td>
<td>fallback</td>
<td>scheme</td>
</tr>
<tr>
<td>scheme</td>
<td>English</td>
<td>scheme</td>
</tr>
<tr>
<td>scheme</td>
<td>German</td>
<td>Schema</td>
</tr>
<tr>
<td>scheme</td>
<td>Norwegian</td>
<td>skjema</td>
</tr>
<tr>
<td>scheme</td>
<td>Nynorsk</td>
<td>skjema</td>
</tr>
<tr>
<td>Scheme</td>
<td>fallback</td>
<td>Scheme</td>
</tr>
<tr>
<td>Scheme</td>
<td>English</td>
<td>Scheme</td>
</tr>
<tr>
<td>Scheme</td>
<td>German</td>
<td>Schema</td>
</tr>
<tr>
<td>Scheme</td>
<td>Norwegian</td>
<td>Skjema</td>
</tr>
<tr>
<td>Scheme</td>
<td>Nynorsk</td>
<td>Skjema</td>
</tr>
<tr>
<td>schemes</td>
<td>fallback</td>
<td>schemes</td>
</tr>
<tr>
<td>schemes</td>
<td>English</td>
<td>schemes</td>
</tr>
<tr>
<td>schemes</td>
<td>German</td>
<td>Schemata</td>
</tr>
<tr>
<td>schemes</td>
<td>Norwegian</td>
<td>skjema</td>
</tr>
<tr>
<td>schemes</td>
<td>Nynorsk</td>
<td>skjema</td>
</tr>
<tr>
<td>Schemes</td>
<td>fallback</td>
<td>Schemes</td>
</tr>
<tr>
<td>Schemes</td>
<td>English</td>
<td>Schemes</td>
</tr>
<tr>
<td>Schemes</td>
<td>German</td>
<td>Schemata</td>
</tr>
<tr>
<td>Schemes</td>
<td>Norwegian</td>
<td>Skjema</td>
</tr>
<tr>
<td>Schemes</td>
<td>Nynorsk</td>
<td>Skjema</td>
</tr>
<tr>
<td>phase-aqi</td>
<td>fallback</td>
<td>aq,$\infty$</td>
</tr>
<tr>
<td>phase-cd</td>
<td>fallback</td>
<td>cd</td>
</tr>
<tr>
<td>phase-lc</td>
<td>fallback</td>
<td>lc</td>
</tr>
</tbody>
</table>

6.5.3. Information For Module Writers

In addition to the commands from section 6.5.2 starting on page 30 the following macros are available:

\texttt{\textbackslash chemmacros\_translate:n \{\textit{translation key}\}}

Translates the given key to the language which is detected automatically or given by the user. Should be used in 	exttt{CHEMMACROS'} macros instead of translations’ \texttt{\textbackslash GetTranslation}.

\texttt{\textbackslash \_chemmacros\_language\_tl}

A token list variable that holds the language which is used by \texttt{\textbackslash chemmacros\_translate:n} for translation, after begin document.

\texttt{\textbackslash ChemTranslate{\{\textit{translation key}\}}}

A version of \texttt{\textbackslash chemmacros\_translate:n} for those who prefer traditional \LaTeX \texttt{2e} programming
6. Internal Modules

over expl3.

\chemmacros_declare_translation:nnn\{\text{language}\}\{\text{key}\}\{\text{translation}\}

The expl3 version of \DeclareChemTranslation.

\chemmacros_declare_translations:nn\{\text{key}\}\{\text{language} = \text{translation}\}

The expl3 version of \DeclareChemTranslations.
Part III.
Additional Modules

7. User Modules

7.1. The all pseudo-module

The all module is a pseudo module: it doesn’t define any functionality at all. It does however load all other modules. So you can say

```
\chemsetup{ modules = all }
```

to ensure that every module is available. This *will not* load personal modules!

7.2. The isotopes Module

The isotopes module loads the elements package [Nie15]. This module defines one user command:

```
\isotope*{(input)}
```

*(input)* can either be the symbol of an element or the name of an element. Be aware that *the name is language dependent*, refer to the manual of the elements package for details. To be on the safe side use the element symbol.

*(input)* can also be comma separated list: \isotope{(nuc),(symbol)}. If you leave *(nuc)* out then \isotope will display the most common isotope. Otherwise *(nuc)* will be used. If *(nuc)* is an isotope unknown to the elements package \isotope will write a warning to the log file.

The starred variant omits the element number.

\begin{verbatim}
1 \isotope(C)
2 \isotope*(C)
3 \isotope(14,C)
4 \isotope*(14,C)
\end{verbatim}

As input for the element symbol you can choose any of the elements known to the elements package.

There are options which allow you to determine how the isotope is printed:

- **isotopes > format** = super|side
  Default: super
  Either print the isotope number as superscript or to the right of the element symbol.

- **isotopes > side-connect** = {(input)}
  Default: -
  Determine what is printed between the element symbol and the isotope number if **format** = {side}.  

\begin{tabular}{ccc}
  $^6_6$C & $^{12}_6$C & $^{14}_6$C \\
\end{tabular}
7. User Modules

7.3. The mechanisms Module

The module `mechanisms` loads the package `amstext` [MSoo]. It provides one macro:

\[ \textbf{\texttt{\textbackslash mech}}[(\textit{type})] \]

Allows to specify the most common reaction mechanisms.

\textit{(type)} can have one of the following values:

\[ \textbf{\texttt{\textbackslash mech}} \]

( empty, no opt. argument) nucleophilic substitution \( S_N \)

\[ \textbf{\texttt{\textbackslash mech}}[1] \]

unimolecular nucleophilic substitution \( S_{N1} \)

\[ \textbf{\texttt{\textbackslash mech}}[2] \]

bimolecular nucleophilic substitution \( S_{N2} \)

\[ \textbf{\texttt{\textbackslash mech}}[se] \]

electrophilic substitution \( S_E \)

\[ \textbf{\texttt{\textbackslash mech}}[1e] \]

unimolecular electrophilic substitution \( S_{E1} \)

\[ \textbf{\texttt{\textbackslash mech}}[2e] \]

bimolecular electrophilic substitution \( S_{E2} \)

\[ \textbf{\texttt{\textbackslash mech}}[ar] \]

electrophilic aromatic substitution \( S_{E} \)

\[ \textbf{\texttt{\textbackslash mech}}[e] \]

elimination \( E \)

\[ \textbf{\texttt{\textbackslash mech}}[e1] \]

unimolecular elimination \( E_1 \)

\[ \textbf{\texttt{\textbackslash mech}}[e2] \]

bimolecular elimination \( E_2 \)

\[ \textbf{\texttt{\textbackslash mech}}[cb] \]

unimolecular elimination “conjugated base”, \( i.e., \) via carbanion \( E_{1cb} \)
7. User Modules

7.4. The newman Module

The newman module provides a command for drawing Newman projections. It loads the tikz module.

\newman[(options)]{(angle){(1), (2), (3), (4), (5), (6)}}

Create Newman projections. This command uses TikZ internally. \(\langle angle\rangle\) rotates the back atoms counter clockwise with respect to the front atoms and is an optional argument. \(\langle 1 \rangle\) to \(\langle 6 \rangle\) are the positions, the first three are the front atoms, the last three the back atoms.

```
\newman{}
\newman(170){}
\newman{1,2,3,4,5,6} \newman{1,2,3} \newman{,,,4,5,6}
```

Several options allow customization:

\texttt{newman \texttt{\textbackslash angle} = \{\langle angle\rangle\}}

Default: 0

Default angle.

\texttt{newman \texttt{\textbackslash scale} = \{\langle factor\rangle\}}

Scale the whole projection by factor \(\langle factor\rangle\).

Default: 1

\texttt{newman \texttt{\textbackslash ring} = \{\langle tikz\rangle\}}

Customize the ring with TikZ keys.

(initially empty)

\texttt{newman \texttt{\textbackslash atoms} = \{\langle tikz\rangle\}}

Customize the nodes within which the atoms are set with TikZ keys.

(initially empty)

\texttt{newman \texttt{\textbackslash back-atoms} = \{\langle tikz\rangle\}}

Explicitly customize the nodes of the back atoms with TikZ keys.

```
\chemsetup[newman]{angle=45} \newman{}
\newman[scale=.75,ring={draw=blue,fill=blue!20}]{}
\chemsetup[newman]{atoms={draw=red,fill=red!20,inner sep=2pt,rounded corners}}
\newman{1,2,3,4,5,6}
```
7. User Modules

7.5. The orbital Module

The orbital module loads the tikz module. It provides the following command to create orbitals:

\orbital[(options)]{(type)}

Draw an orbital shape of type \textit{(type)}. This command uses Ti\kZ\ internally.

There are the following types available for \textit{(type)}:

\begin{itemize}
    \item s
    \item p
    \item sp
    \item sp2
    \item sp3
\end{itemize}

Depending on the type you have different options to modify the orbitals:

\begin{itemize}
    \item \texttt{orbital\textunderscore phase = \{\textit{factor}\}} \hspace{1cm} \text{Default: +}
        \text{changes the phase of the orbital (all types)}
    \item \texttt{orbital\textunderscore scale = \{\textit{factor}\}} \hspace{1cm} \text{Default: 1}
        \text{changes the size of the orbital (all types)}
    \item \texttt{orbital\textunderscore color = \{\textit{color}\}} \hspace{1cm} \text{Default: black}
        \text{changes the color of the orbital (all types)}
\end{itemize}
7. User Modules

\textbf{orbital \_\_ angle = \{\langle angle \rangle\}}  \hspace{1cm} \text{Default: 0}
rotates the orbitals with a p contribution counter clockwise (all types except s)

\textbf{orbital \_\_ half = true|false} \hspace{1cm} \text{Default: false}
displays only half an orbital (only p)

Additionally there are two options, with which the Ti\kZ behaviour can be changed.

\textbf{orbital \_\_ overlay = true|false}
The orbital “doesn’t need space”; it is displayed with the Ti\kZ option overlay.

\textbf{orbital \_\_ opacity = \{\langle num \rangle\}}
The orbital becomes transparent; \langle value \rangle can have values between 1 (fully opaque) to 0 (invisible).

\begin{verbatim}
1 \vspace{7mm}
2 \chemsetup{orbital}{
3   overlay,
4   p/color = black!70
5 }
6 \setchemfig{bond offset=0pt}
7 \chemfig{
8   ?\orbital{p}
9   -[,1.3]{\orbital[phase=-]{p}}
10   -[:30,1.1]{\orbital{p}}
11   -[:150,.9]{\orbital[phase=-]{p}}
12   -[4,1.3]{\orbital{p}}
13   -[:{-150},1.1]{\orbital[phase=-]{p}}?
14 }
15 \vspace{7mm}
\end{verbatim}
\begin{verbatim}
\vspace{7mm}
\setchemfig{bond offset = 0pt}
\chemsetup[orbital]{
  overlay ,
  opacity = .75 ,
  p/scale = 1.6 ,
  s/color = blue!50 ,
  s/scale = 1.6
}
\chemfig{
  \orbital{s} -\[:-20\]{\orbital[angle=0]{p}}
  \orbital[half,angle=0]{p}
  \orbital[angle=170,angle=170,angle=170,angle=170,angle=170]{p}
  \orbital[angle=-150,angle=-150,angle=-150,angle=-150,angle=-150]{p}
  (-\[:-150\]{\orbital{s}}) -\orbital{s}
}
\vspace{1cm}
\end{verbatim}

\subsection{The polymers Module}

The polymers module loads the nomenclature and the \texttt{tikz} modules.

\subsubsection{Nomenclature}

The polymers module defines a number of \texttt{IUPAC} macros for usage inside \texttt{\iupac} which are used in polymer chemistry.

\textbf{Copolymers}

\verbatim{copolymers}{co}

unspecified copolymer. An alias for this command is \texttt{co}.

\verbatim{statistical}{stat}

statistical copolymer. An alias for this command is \texttt{stat}.
\random \ ran
random copolymer. An alias for this command is \ran.

\alternating \ alt
alternating copolymer. An alias for this command is \alt.

\periodic \ per
periodic copolymer. An alias for this command is \per.

\block \ block
block copolymer.

\graft \ graft
graft copolymer.

Non-linear (Co) Polymers and Polymer Assemblies

\blend \ blend
The blend qualifier.

\comb \ comb
The comb qualifier.

\complex \ compl
The complex qualifier. An alias for this command is \compl.

\cyclic \ cyclo
The cyclic qualifier. An alias for this command is \cyclo.

\branch \ branch
The branch qualifier.

\network \ net
The network qualifier. An alias for this command is \net.

\ipnetwork \ ipn
The interpenetrating network qualifier. An alias for this command is \ipn.

\sipnetwork \ sipn
The semi-interpenetrating network qualifier. An alias for this command is \sipn.

\star \ star
The star qualifier.

7.6.2. Polymer Denotations in \textit{chemfig}'s Molecules

The \textit{chemfig} manual proposes some code defining the macros \setpolymerdelim and \makebraces
which make it possible to add delimiters to \textit{chemfig} molecules. The \polymers module imple-
ments the following macro based on the same idea:

\makepolymerdelims\{(options)\}\{\height\}\{\depth\}\{\opening node\}\{\closing node\}\}

The value of \textit{depth} is the same as \textit{height} unless it is specified explicitly. \textit{opening node} and
\textit{closing node} are the names of \textit{TikZ} nodes where the delimiters are placed.
7. User Modules

polymers \( \texttt{delimiters} = \{ \left( \right) \} \)

This option demands two tokens as argument, the first being the opening brace, the second the closing brace. A dot (.) denotes an empty delimiter.

polymers \( \texttt{subscript} = \{ \texttt{subscript} \} \)

Subscript to the right delimiter.

polymers \( \texttt{superscript} = \{ \texttt{superscript} \} \)

Superscript to the right delimiter.

\begin{verbatim}
1 \setchemfig{atom sep=2em}
2 \chemfig{[-@\text{(op,.75)}]CH_2-CH(-[6]Cl)-[@\text{cl,0.25}]})}
3 \makepolymerdelims{5pt}{27pt}{op}{cl}
4 \chemfig{[-@\text{(op,.75)}]CH_2-CH(-[6]Cl)-[@\text{cl,0.25}]})}
5 \makepolymerdelims{delimiters=}{}{5pt}{27pt}{op}{cl}
\end{verbatim}

\[
\begin{array}{l}
\text{CH}_2-\text{CH} \\
\mid \text{Cl} \\
\mid \text{Cl} \\
\end{array}
\quad n
\begin{array}{l}
\text{CH}_2-\text{CH} \\
\mid \text{Cl} \\
\mid \text{Cl} \\
\end{array}
\]

7.7. The reactions Module

The reactions module loads the chemformula module and the mathtools package [MRW19].

7.7.1. Predefined Environments

You can use these environments for numbered...

\begin{verbatim}
\begin{reaction}
A single reaction where \texttt{CHEMFORMULA} code is placed directly in the environment body. A wrapper around the equation environment. The environment body is parsed with \texttt{ch} or \texttt{ce} depending on the value of the \texttt{formula} option, see section 6.2 starting on page 27.
\end{reaction}
\begin{verbatim}
\begin{reaction*}
A wrapper around the \texttt{equation*} environment. The environment body is parsed with \texttt{ch} or \texttt{ce} depending on the value of the \texttt{formula} option, see section 6.2 starting on page 27.
\end{reaction*}
\end{verbatim}
\end{verbatim}

... and their starred versions for unnumbered reactions.

\begin{verbatim}
\begin{reactions}
Several aligned reactions. A wrapper around amsmath’s \texttt{align} environment. The environment body is parsed with \texttt{ch} or \texttt{ce} depending on the value of the \texttt{formula} option, see section 6.2 starting on page 27.
\end{reactions}
\begin{verbatim}
\begin{reactions*}
A wrapper around amsmath’s \texttt{align*} environment. The environment body is parsed with \texttt{ch} or \texttt{ce} depending on the value of the \texttt{formula} option, see section 6.2 starting on page 27.
\end{reactions*}
\end{verbatim}
\end{verbatim}

With those environments you can create (un)numbered reaction equations similar to mathematical equations.

Theses environments use the \texttt{equation/equation*} environments or the \texttt{align/align*} environments, respectively, to display the reactions.
If you want to change the layout of the counter tags, you can use
\renewtagform{tagname}{format}{left delimiter}{right delimiter}
Provided by the mathtools package.

\texttt{\textbackslash reactions} » \texttt{tag-open} = {{\texttt{left delimiter}}} \\
\texttt{The left delimiter.}
7. User Modules

**reactions** » **tag-close** = {{right delimiter}}

The right delimiter.

**reactions** » **before-tag** = {{format}}

Code inserted before the tags.

```
\chemsetup[reactions]{
  before-tag = R \textbf{, 
  tag-open = [ ,
  tag-close = ]
}
\begin{reaction}
H2O + CO2 ⇌ H2CO3
\end{reaction}
```

H₂O + CO₂ ⇌ H₂CO₃

The use of \LaTeX's \textit{\texttt{intertext}} is possible:

```
\begin{reaction}
A + 2 B \rightarrow 3 C + D \ "\textit{label}\{rxn: test\}"
\textit{intertext}{Some text in between aligned reactions}
3 E + F \leftrightarrow G + \frac{1}{2} H
\end{reaction}
See reaction-\textit{\texttt{ref}}\{rxn: test\}.
```

A + 2 B → 3 C + D

Some text in between aligned reactions

3 E + F ↔ G + \frac{1}{2} H

See reaction 5.

If you are using either cleveref or fancyref the reaction counter is supported already. For fancyref use the prefix \texttt{rct}.

7.7.2. Own Reactions

You can create new types of reactions with the command:

`\textit{NewChemReaction}\{\langle name\rangle\}\{\langle number of arguments\rangle\}\{\langle math name\rangle\}`

\langle name\rangle will be the name of the new chem environment. \langle math name\rangle is the underlying math environment. Gives an error if \langle name\rangle already exists.

`\textit{RenewChemReaction}\{\langle name\rangle\}\{\langle number of arguments\rangle\}\{\langle math name\rangle\}`

\langle name\rangle is the name of the renewed chem environment. \langle math name\rangle is the underlying math environment. Gives an error if \langle name\rangle does not exist.
7. User Modules

\DeclareChemReaction{(name)}{(number of arguments)}{(math name)}

(name) will be the name of the chem environment. (math name) is the underlying math environment.

\ProvideChemReaction{(name)}{(number of arguments)}{(math name)}

(name) will be the name of the new chem environment. (math name) is the underlying math environment. The new environment is only defined if it doesn’t exist, yet.

\NewChemReaction{reaction} {equation}
\NewChemReaction{reaction*} {equation*}
\NewChemReaction{reactions} {align}
\NewChemReaction{reactions*}{align*}

Let’s suppose, you’d like to have the alignment behaviour of the alignat environment for \texttt{CHEMFORMULA} reactions. You could do the following:

\NewChemReaction{reactionsat}[1]{alignat}

With this the \texttt{reactionsat} environment is defined.

\begin{reactionsat}{3}
A &-> B &-> C &-> D \\
aaaaa &-> bbbbb &-> ccccc &-> ddddd
\end{reactionsat}

\begin{reactionsat*}{2}
A &-> B & C &-> D \\
aaaaa &-> bbbbb & \quad{} ccccc &-> ddddd
\end{reactionsat*}

\begin{itemize}
\item \texttt{A \rightarrow B} \quad \texttt{C \rightarrow D} \quad \texttt{\{7\}}
\item \texttt{aaaaa \rightarrow bbbbb} \quad \texttt{cccccc \rightarrow ddddd} \quad \texttt{\{8\}}
\end{itemize}

\begin{itemize}
\item \texttt{A \rightarrow B} \quad \texttt{C \rightarrow D}
\item \texttt{aaaaa \rightarrow bbbbb} \quad \texttt{cccccc \rightarrow ddddd}
\end{itemize}

### 7.7.3. List of Reactions

The \texttt{reactions} module also provides a command to display a list of the reactions created with the \texttt{reaction} environment.

\listofreactions

Print a list of reactions.
7. User Modules

---

List of Reactions

Reaction {1}  
Reaction {2}  
Reaction {3}  
Reaction [R 4]  
Reaction {5}  
Reaction {6}  
Reaction {7}  
Reaction {8}  
Reaction {9}: Autoprotolyse  
Reaction {10}: first step of chain  
Reaction {11}: second step of chain

The output of this list can be modified by two options:

- **list-name** = {{name of the list}}  
  Default: \ChemTranslate{list-of-reactions}  
  Let’s you set the name of the list manually. The default name is language dependent, see section 6.5 starting on page 30.

- **list-entry** = {{prefix to each entry}}  
  Default: \ChemTranslate{reaction}  
  Let’s you set a prefix to each list entry. The default name is language dependent, see section 6.5 starting on page 30.

- **list-heading-cmd** = {{code}}  
  Default: \section*{#1}  
  The macro that is called at the beginning of the list. Inside of {{code}} #1 refers to the actual heading of the list. The default setting is not entirely true: if a macro \chapter is defined \chapter*{#1} is used.

- **tocbasic** = true|false  
  Default: false  
  If you use a KOMA-Script class or if you load the tocbasic package or if you set this option to true the list of reactions will be set up using the tocbasic package. This disables the list-heading-cmd option. For a KOMA-Script class this means that the list of reactions obeys KOMA-Script’s listof option.

Instead of using the option list-name you also could redefine \reactionlistname.

The list lists all reactions with a number and disregards reactions without number. All reaction environments without star have an optional argument which let’s you add a description (or caption) for the entry in the list.

\begin{reaction}{Autoprotolyse}
\begin{reaction}
2 H2O $\rightleftharpoons$ H3O$^+$ + OH$^-$
\end{reaction}

2H₂O $\rightleftharpoons$ H₃O⁺ + OH⁻  

If you use the reactions environment this will not work, though. In this case you can use
\AddRxnDesc{(description)}
Add a description to a reaction.

\begin{reactions}
  "\chlewis{0.}{Cl}" + CH4 & -> HCl + "\chlewis{180.}{C}" H3 \AddRxnDesc{first-step-of-chain} \\
  "\chlewis{180.}{C}" H3 + Cl2 & -> CH3Cl + "\chlewis{0.}{Cl}" \AddRxnDesc{second-step-of-chain}
\end{reactions}

\[\text{Cl}^\text{•} + \text{CH}_4 \rightarrow \text{HCl} + \text{•CH}_3\] \[\text{CH}_3\text{Cl} + \text{Cl}^\text{•}\] \text{(10)} \text{(11)}

### 7.1. The \texttt{redox} Module

The \texttt{redox} module loads the modules tikz and xfrac. It also loads the packages math- tools [MRW19] and relsize [Ars13].

#### 7.1.1. Oxidation Numbers

Regarding the typesetting of oxidation numbers *The IUPAC Green Book* [Coh+08] says the following:

Oxidation numbers are denoted by positive or negative Roman numerals or by zero […]

Examples Mn\textsuperscript{VII}, manganese (VII), O\textsuperscript{II}, Ni\textsuperscript{0} \text{[Coh+08, p. 50]}

The following command is provided to set oxidation numbers:

\texttt{\textbackslash ox\{\textbackslash options\}\{\textbackslash number\},\textbackslash atom\}}

Places \texttt{\textbackslash number} as right superscript to \texttt{\textbackslash atom}; \texttt{\textbackslash number} has to be a (rational) number! \texttt{\textbackslash atom} is treated as a \texttt{chemformula} formula, like it would be in \texttt{\chcpd} (this depends on the setting of the \texttt{formula} option, see 6.2 starting on page 27).

\[\text{Na}^\text{I}, \text{Ca}^\text{II}, \text{S}^\text{II}, \text{F}^\text{I}\]

There are a number of options that can be used to modify the typeset result:

- \texttt{\textbackslash redox\textbackslash format = \{(\textbackslash code)\}}
  
  Allows to apply arbitrary \texttt{\textbackslash code} in front of the typeset oxidation numbers. The last command may expect the oxidation number as an argument. An example might be \texttt{\textcolor{red}}.

- \texttt{\textbackslash redox\textbackslash parse = true|false}
  
  Default: true
  
  When \texttt{false} an arbitrary entry can be used for \texttt{\textbackslash number}.

- \texttt{\textbackslash redox\textbackslash roman = true|false}
  
  Default: false
  
  Switches from roman to arabic numbers.
redox » \pos = \{top|super|side\} Default: super
    top places \\(\text{\textit{number}}\) above \\(\text{\textit{atom}}\), super to the upper right as superscript and side to the right and inside brackets. Both super and side follow IUPAC recommendation, top does not!

\[ \text{\textit{explicit-sign}} = \text{\textit{true|false}} \]
    Shows the + for positiv numbers and the ± for 0.

\[ \text{\textit{explicit-zero-sign}} = \text{\textit{true|false}} \]
    Default: true
    Only if both \text{\textit{explicit-sign}} and \text{\textit{explicit-zero-sign}} are set to true ±0 will be printed.

\[ \text{\textit{decimal-marker}} = \text{\textit{comma|point}} \]
    Default: point
    Choice for the decimal marker for formal oxidation numbers like \(X^{1.2}\).

redox » \text{\textit{align}} = \text{\textit{center|right}}
Center the oxidation number relative to the atom or right-align it.

\[ \text{\textit{side-connect}} = \{\text{\textit{code}}\} \]
    Default: \, Code that is inserted between atom and oxidation number if \pos = \{side\} is used.

\[ \text{\textit{text-frac}} = \{\text{\textit{cs}}\} \]
    Default: \chemfrac\{text\}{\#1}{\#2}
    The fraction macro that is used for fractions if \pos = \{side\} is used. \{cs\} must be a macro that takes two mandatory arguments, the first for the numerator and the second for the denominator.

\[ \text{\textit{super-frac}} = \{\text{\textit{cs}}\} \]
    Default: \chemfrac\{superscript\}{\#1}{\#2}
    The fraction macro that is used for fractions if \pos = \{top\} or \pos = \{super\} is used. \{cs\} must be a macro that takes two mandatory arguments, the first for the numerator and the second for the denominator.

\begin{verbatim}
1 \ox[roman=false]{2,Ca} \ox{2,Ca} \ Ca^2 \ Ca^{II} \\
2 \ox[pos=top]{3,Fe}\text{-Oxide} \ Fe-Oxide \\
3 \ox[pos=side]{3,Fe}\text{-Oxide} \ Fe\,(III)\text{-Oxide} \\
4 \ox[parse=false]{{?,Mn}} \ Mn^{II} \\
5 \ox[pos=top,align=right]{2,Ca} \ Ca^{III} \\
\end{verbatim}

The \pos = \{top\} variant also can be set with the shortcut \ox*:

\begin{verbatim}
1 \ox{3,Fe} \ox*(3,Fe) \ \ Fe^{III} \ Fe \\
\end{verbatim}

Using the \text{\textit{explicit-sign}} option will always show the sign of the oxidation number:

\begin{verbatim}
1 \chemsetup[redox]{explicit-sign = true} \ox{+1,Na}, \ox{2,Ca}, \ox{-2,S}, \ch{\"\ox{0,F}\"} {}2 \\
\end{verbatim}

\[ \text{\textit{Na}^{II}, \text{\textit{Ca}^{II}}, \text{\textit{S}^{II}}, \text{\textit{F}^{0}}} \]

\begin{verbatim}
1 \chemsetup[redox]{pos=top} \Compare \ox{-1,0} to \ch{\"\ox{-1,0}\"} {}2^2\text{-} \\
\end{verbatim}
Compare $O_2^{+1}$ to $O_2^{-1}$.

Sometimes one might want to use formal oxidation numbers like 0.5 or $\frac{1}{3}$:

\begin{verbatim}
\chemsetup[redox]{pos=top}
\ox{0.5,Br2}
\ch{"\ox{1/3,I}" {3+}}
\chemsetup[redox]{pos=side}
\ox{1/3,I3+}
\end{verbatim}

The fraction is displayed with the help of the xfrac package \cite{L3Pb}. For more details on how \textsc{chemmacros} uses it read section 8.2 starting on page 64.

\subsection{Redox Reactions}

\textsc{chemmacros} provides two commands to visualize the transfer of electrons in redox reactions. Both commands are using Ti\textsf{k}Z.

\begin{verbatim}
\OX{(name),\{atom\}}
\redox{(name1),\{name2\}\{\langle tikz\rangle\{\langle num\rangle\}\{\langle text\rangle\}}
\end{verbatim}

Label \{atom\} with the label \{name\}.

Connect two \{atom\}'s previously labelled with \texttt{\OX}. Only the first argument (\langle name1\rangle,\langle name2\rangle) is required, the others are all optional.

\texttt{\OX} places \{atom\} into a node, which is named with \{name\}. If you have set two \texttt{\OX}, they can be connected with a line using \texttt{\redox}. To do so the names of the two nodes that are to be connected are written in the round braces. Since \texttt{\redox} draws a \texttt{tikzpicture} with options remember picture, overlay, the document needs to be \textit{compiled at least two times}.

\begin{verbatim}
\vspace{7mm}
\OX{a,Na} $\rightarrow$ \OX{b,Na}\pch\redox(a,b){oxidation}
\end{verbatim}

This line can be customized using Ti\textsf{k}Z keys in \langle \texttt{tikz} \rangle:

\begin{verbatim}
\vspace{7mm}
\OX{a,Na} $\rightarrow$ \OX{b,Na}\pch\redox(a,b)\rightarrow,red\{ox\}
\end{verbatim}

With the argument \langle num \rangle the length of the vertical parts of the line can be adjusted. The default length is \texttt{.6em}. This length is multiplied with \langle num \rangle. If you use a negative value the line

48
The default length of the vertical lines can be customized with the option
\begin{verbatim}
\texttt{\redox\dist = \{(dim)\}} \quad \text{Default: .6em}
\end{verbatim}
A \TeX{} dimension.
\begin{verbatim}
\texttt{\chemsetup{redox/dist=1em}}
\end{verbatim}

\begin{verbatim}
\texttt{\OX{a,Na} \rightarrow \OX{b,Na}\redox(a,b)[->,red]{ox}}
\end{verbatim}

\begin{verbatim}
\texttt{Na \to Na^+}
\end{verbatim}

The option can be used to change the distance between the atom and the beginning of the line.
\begin{verbatim}
\texttt{\chemsetup{redox/sep=.5em}}
\end{verbatim}

\begin{verbatim}
\texttt{\OX{a,Na} \rightarrow \OX{b,Na}\redox(a,b)[->,red]{ox}}
\end{verbatim}

\begin{verbatim}
\texttt{Na \to Na^+}
\end{verbatim}

**Examples**

\begin{verbatim}
\texttt{\ch{2 "\OX{o1,Na}" + "\OX{r1,Cl}" \{}2
  \rightarrow
  2 "\OX{o2,Na}" \{} + + 2 "\OX{r2,Cl}" \{}.
}}
\end{verbatim}

\begin{verbatim}
\texttt{\redox(o1,o2)\small: $-$ 2\el$}$
\end{verbatim}

\begin{verbatim}
\texttt{\redox(r1,r2)\small: $+$ 2\el$} \quad \texttt{\vspace{7mm}}
\end{verbatim}
7. User Modules

\[
\begin{align*}
\text{OX: } & -2e^- \\
2 \text{Na} + \text{Cl}_2 & \rightarrow 2 \text{Na}^+ + 2 \text{Cl}^- \\
\text{RED: } & +2e^-
\end{align*}
\]
7. User Modules

7.2. The scheme Module

The scheme module loads the chemnum package [Nie16b] and defines a floating environment \begin{scheme}. That is, it only defines this float if no environment \texttt{scheme} exists at the end of the preamble. The module checks for different available float defining methods, in this order:

- If the current class is a KOMA-Script class \texttt{DeclareNewTOC} will be used.
- If the current class is memoir, memoir’s methods are used.
- If the package tocbasic has been loaded \texttt{DeclareNewTOC} will be used.
- If the package newfloat has been loaded \texttt{DeclareFloatingEnvironment} will be used.
- If the package floatrow has been loaded its method will be used.
- If the package float has been loaded its method will be used.
- If neither of the above the “manual” method is used. This means the environment is defined the same way like figure is defined in the article class or the book class, depending if \texttt{chapter} is defined or not.

The list name and the caption name both are translated to the language specified according to the \texttt{lang} option and the provided translations, see section 6.5 starting on page 30 for details. If you want to manually change them then redefine these macros after begin document:

\begin{itemize}
  \item \texttt{listschemename} The name of the list of schemes.
  \item \texttt{schemename} The name used in captions.
  \item The list of schemes is printed as expected with \texttt{listofschemes}
\end{itemize}

If you are using either cleveref or fancyref the scheme environment (or rather its captions) are supported already. For fancyref use the prefix \texttt{sch}.

7.3. The spectroscopy Module

The spectroscopy module loads the chemformula module and the siunitx package [Wri18].
7. User Modules

7.3.1. The \texttt{NMR} Command

When you’re trying to find out if a compound is the one you think it is often NMR spectroscopy is used. The experimental data are typeset similar to this:

\[ ^1\text{H}-\text{NMR} (400 \text{ MHz}, \text{CDCl}_3): \delta = 1.59 \]

The spectroscopy module provides a command which simplifies the input.

\texttt{NMR*((num), (element))((num), (unit))[(solvent)]}

Typeset nuclear magnetic resonance data. \textit{(num)} is a valid \texttt{siunitx} number input, \textit{(unit)} is a valid \texttt{siunitx} unit input. \textit{(solvent)} is any valid \texttt{CHEMFORMULA} input as in \texttt{\chcpd} (this depends on the setting of the \texttt{formula} option, see 6.2 starting on page 27).

\textit{All Argument are optional! Without arguments we get:}

\begin{itemize}
\item \texttt{NMR} \par
\item \texttt{NMR*}
\end{itemize}

The first argument specifies the kind of NMR:

\begin{itemize}
\item \texttt{NMR(13,C)}
\item \texttt{NMR(400)}
\end{itemize}

The second argument sets the frequency (in MHz):

\begin{itemize}
\item \texttt{NMR(4e8,\texttt{hertz})}
\end{itemize}

You can choose another unit:

\begin{itemize}
\item \texttt{sisetup\{exponent-product=\texttt{\cdot}\}}
\item \texttt{NMR(4e8,\texttt{hertz})}
\end{itemize}

Please note that the setup of \texttt{siunitx} also affects this command:

\begin{itemize}
\item \texttt{NMR[CDCl3]}
\end{itemize}

The third argument specifies the solvent:

\begin{itemize}
\item \texttt{\textbf{Short Cuts}}
\item It is possible to define short cut commands for specific nuclei.
\item \texttt{NewChemNMR\{(cs)\}((num), (atom)\)}
\item Define a new shortcut macro for typesetting a certain type of magnetic resonance data. Gives an error if \textit{\textit{\textit{cs}}} already exists.
\end{itemize}
7. User Modules

\DeclareChemNMR{\{cs\}\{\langle num\rangle,\langle atom\rangle\}}
Define a new shortcut macro for typesetting a certain type of magnetic resonance data. Over-
writes an existing macro.

\RenewChemNMR{\{cs\}\{\langle num\rangle,\langle atom\rangle\}}
Redefine an existing shortcut macro for typesetting a certain type of magnetic resonance data.
Gives an error if \{cs\} doesn’t exist.

\ProvideChemNMR{\{cs\}\{\langle num\rangle,\langle atom\rangle\}}
Define a new shortcut macro for typesetting a certain type of magnetic resonance data. \{cs\} is
only defined if it doesn’t exist, yet.

This defines a command with the same arguments as \NMR except for \{\langle num\rangle,\langle atom\rangle\}:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>\NewChemNMR\HNMR{1,H}%</td>
<td>13C-NMR (100 MHz)</td>
</tr>
<tr>
<td>2</td>
<td>\NewChemNMR\CNMR{13,C}%</td>
<td>1H-NMR (400 MHz)</td>
</tr>
<tr>
<td>3</td>
<td>\CNMR*(100) \par</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>\HNMR*(400)</td>
<td></td>
</tr>
</tbody>
</table>

7.3.3. An Environment to Typeset Experimental Data

The spectroscopy module provides an environment to ease the input of experimental data.

\begin{experimental}
Environment for the output of experimental data. Inside the environment the following com-
mmands are defined.

\data{\{type\}\{\langle specification\rangle\}}
Type of data, e.g. IR, MS... The optional argument takes further specifications which are output
in parentheses.

\data*{\{type\}\{\langle specification\rangle\}}
Like \data but changes the = into a : , given that use-equal = \{true\} is used.

\NMR{\{\langle num\rangle,\langle elem\rangle\}\{\langle coupling core\rangle\}\{\langle num\rangle,\langle unit\rangle\}\{\langle solvent\rangle\}}
This command gets an additional argument: \NMR{13,C[^1H]} 13C[^1H]-NMR: δ

\(*\{\langle bonds\rangle;\langle nuclei\rangle\}\{\langle unit\rangle\}\{\langle list of nums\rangle\}
Coupling constant, values are input separated by ; (NMR). The arguments \{\langle bonds\rangle;\langle nuclei\rangle\}
and \{\langle unit\rangle\} are optional and enable further specifications of the coupling.

\#{\{\langle num\rangle\}}
Number of nuclei (NMR).

\pos{\{\langle num\rangle\}}
Position of nuclues (NMR).

\val{\{\langle num\rangle\}}
A number, an alias of siunitx’ \num{\{\langle num\rangle\}}.

\val{\{\langle num1\rangle\} - \{\langle num2\rangle\}}
An alias of siunitx’ \numrange{\{\langle num1\rangle\}}{\{\langle num2\rangle\}}.

53
**Customization**  The output of the environment and of the NMR commands can be customized be a number of options. For historical reasons they all belong to the module `nmr`.

```latex
\begin{experimental}
data{type1} Data.
data{type2} [specifications] More data.
data*{type3} Even more data.
\end{experimental}
```

**spectroscopy** » unit = \{(unit)\} \hspace{1cm} Default: \texttt{\textbackslash mega\hertz}

The used default unit.

**spectroscopy** » nucleus = \{(num),\{atom\}\} \hspace{1cm} Default: \{1,H\}

The used default nucleus.

**spectroscopy** » connector = \{(code)\} \hspace{1cm} Default: -

Places \{code\} between the nucleus and the method.

**spectroscopy** » method = \{(code)\} \hspace{1cm} Default: NMR

The measuring method.

**spectroscopy** » format = \{(commands)\} \hspace{1cm} (initially empty)

For example `\bfseries`.

**spectroscopy** » nmr-base-format = \{(commands)\} \hspace{1cm} (initially empty)

Formatting instructions for the NMR base.

**spectroscopy** » pos-number = \{side\sub\super\} \hspace{1cm} Default: \texttt{side}

Position of the number next to the atom.

**spectroscopy** » coupling-symbol = \{(code)\} \hspace{1cm} Default: J

The symbol used for the coupling constant.

**spectroscopy** » coupling-unit = \{(unit)\} \hspace{1cm} Default: \texttt{\textbackslash hertz}

A \texttt{siunitx} unit.

**spectroscopy** » coupling-pos = \{side\sub\} \hspace{1cm} Default: \texttt{side}

Placement of the coupling nuclei next to the symbol \texttt{J} (or rather the symbol specified with option `coupling-symbol`).

**spectroscopy** » coupling-nuclei-pre = \{(code)\} \hspace{1cm} Default: ( \texttt{}} \texttt{)}

Code inserted before the coupling nuclei when `coupling-pos = \{side\}`.

**spectroscopy** » coupling-nuclei-post = \{(code)\} \hspace{1cm} Default: ) \texttt{}} \texttt{)}

Code inserted after the coupling nuclei when `coupling-pos = \{side\}`.

**spectroscopy** » coupling-bonds-pre = \{(code)\} \hspace{1cm} (initially empty)

Code inserted before the coupling bonds.

**spectroscopy** » coupling-bonds-post = \{(code)\} \hspace{1cm} Default: \texttt{\!}

Code inserted after the coupling bonds.
spectroscopy » coupling-pos-cs = \{\texttt{cs}\}\) \hspace{1cm} \text{Default: \texttt{@firstofone}}

Set the macro that prints the number set with the \texttt{pos} macro. This needs to be a command with one mandatory argument.

spectroscopy » atom-number-cs = \{\texttt{cs}\}\) \hspace{1cm} \text{Default: \texttt{@firstofone}}

Set the macro that prints the number set with the \texttt{#} macro. This needs to be a command with one mandatory argument.

spectroscopy » atom-number-space = \{\texttt{dim}\}\) \hspace{1cm} \text{Default: \texttt{.16667em}}

Introduced in version 5.3 Horizontal space inserted between number and atom (printed by \texttt{#}).

spectroscopy » parse = true|false \hspace{1cm} \text{Default: true}

treat the solvent as \texttt{chemformula} formula (this depends on the setting of the \texttt{formula} option, see 6.2 starting on page 27) or not.

spectroscopy » delta = \{\texttt{tokens}\}\) \hspace{1cm} \text{(initially empty)}
The \texttt{\textit{tokens}} are added after $\delta$.

spectroscopy » list = true|false \hspace{1cm} \text{Default: false}
The environment \texttt{nmr} is formatted as a list

spectroscopy » list-setup = \{\texttt{setup}\}\) \hspace{1cm} \text{Setup of the list. See below for the default settings.}

spectroscopy » use-equal = true|false \hspace{1cm} \text{Default: false}
Add equal sign after \texttt{\textbf{NMR}} and \texttt{\textit{data}}.

The default setup of the list:

\begin{verbatim}
\topsep\z@skip \partopsep\z@skip
\itemsep\z@ \parsep\z@ \itemindent\z@
\leftmargin\z@
\end{verbatim}

\begin{verbatim}
\begin{experimental}[format=\bfseries]
\data{type1} Data.
\data{type2}[\textit{specifications}] More data.
\data*{type3} Even more data.
\end{experimental}
\end{verbatim}

\texttt{\textit{Data. \texttt{\textit{type2 \textit{specifications}}} More data. \texttt{\textit{type3}}} Even more data.}

The command \texttt{\textit{NMR}} and all commands defined through \texttt{\textit{NewChemNMR}} can be used like \texttt{\textit{data}} for the NMR data.

\begin{verbatim}
\begin{experimental}[format=\bfseries,use-equal]
\data{type1} Data.
\data{type2}[\textit{specifications}] More data.
\texttt{\textit{NMR}} Even more data.
\end{experimental}
\end{verbatim}
An Example  The code below is shown with different specifications for (options). Of course options can also be chosen with \chemsetup.

```latex
\sisetup{separate-uncertainty, per-mode=symbol, detect-all, range-phrase=--}
\begin{experimental}[<optionen>]
data*{yield} \SI{17}{\milli\gram} yellow needles (\SI{0.04}{\milli\mole}, \SI{13}{\percent}).
data{mp.} \SI{277}{\celsius} (DSC).
\NMR(600)[CDCl3] \val{2.01} (s, \#{24}, \pos{5}), \val{2.31} (s, \#{12}, \pos{1}), \val{6.72--6.74} (m, \#{2}, \pos{11}), \val{6.82} (s, \#{8}, \pos{8}), \val{7.05--7.07} (m, \#{2}, \pos{12}), \val{7.39--7.41} (m, \#{4}, \pos{9}), \val{7.48--7.49} (m, \#{4}, \pos{8}).
\NMR{13,C}(150)[CDCl3] \val{21.2} ($+$, \#{4}, \pos{1}), \val{23.4} ($+$, \#{2}, \pos{5}), \val{126.0} ($+$, \#{4}, \pos{9}), \val{128.2} ($+$, \#{8}, \pos{5}), \val{130.8} ($+$, \#{2}, \pos{12}), \val{133.6} ($+$, \#{2}, \pos{11}), \val{137.0} (q, \#{4}, \pos{10}), \val{140.6} (q, \#{2}, \pos{10}), \val{140.8} (q, \#{8}, \pos{4}), \val{141.8} (q, \#{4}, \pos{6}), \val{145.6} (q, \#{2}, \pos{7}).
\data{MS}[DCP, EI, \SI{60}{\electronvolt}] \val{703} (2, \ch{M+}), \val{582} (1), \val{462} (13), \val{120} (41), \val{105} (100).
\data{MS}[\ch{MeOH + H2O + KI}, ESI, \SI{10}{\electronvolt}] \val{720} (100, \ch{M+ + OH-}), \val{368} (100, \ch{M+ + 2 OH-}).
\data{IR}[KBr] \val{3443} (w), \val{3061} (w), \val{2957} (m), \val{2918} (m), \val{2856} (w), \val{2729} (w), \val{1725} (w), \val{1606} (s), \val{1592} (s), \val{1545} (w), \val{1446} (m), \val{1421} (m), \val{1402} (m), \val{1357} (w), \val{1278} (w), \val{1238} (s), \val{1214} (s), \val{1172} (s), \val{1154} (m), \val{1101} (w), \val{1038} (w), \val{979} (m), \val{874} (m), \val{846} (s), \val{818} (w), \val{798} (w), \val{744} (w), \val{724} (m), \val{663} (w), \val{586} (w), \val{562} (w), \val{515} (w).
\data{UV-Vis} \SI{386}{\nano\metre} ($\varepsilon = \val{65984}$), \SI{406}{\nano\metre} ($\varepsilon = \val{65378}$).
\data{quantum yield} $\Phi = \val{0.74+-0.1}$
\end{experimental}
```

Nearly Standard  Output with these options:
yield: 17 mg yellow needles (0.04 mmol, 13%). mp. = 277 °C (DSC). $^1$H-NMR (600 MHz, CDCl$_3$): $\delta$ (ppm) = 2.01 (s, 24 H, $H_\text{q}$), 2.31 (s, 12 H, $H_\text{s}$), 6.72–6.74 (m, 2 H, $H_{11}$), 6.82 (s, 8 H, $H_9$), 7.05–7.07 (m, 2 H, $H_{12}$), 7.39–7.41 (m, 4 H, $H_8$), 7.48–7.49 (m, 4 H, $H_6$). $^{13}$C-NMR (150 MHz, CDCl$_3$): $\delta$ (ppm) = 21.2 (+, 4 C, $C_1$), 23.4 (+, 8 C, $C_9$), 126.0 (+, 4 C, $C_9$), 128.2 (+, 4 C, $C_9$), 132.8 (+, 2 C, $C_{11}$), 137.0 (+, 4 C, $C_{12}$), 138.6 (q, 4 C, $C_2$), 140.6 (q, 2 C, $C_{10}$), 140.8 (q, 8 C, $C_4$), 141.8 (q, 4 C, $C_5$), 145.6 (q, 2 C, $C_7$). MS (DCP, EI, 60 eV) = 703 (2, M$^+$), 582 (1), 462 (1), 249 (13), 120 (41), 105 (100). MS (MeOH + H$_2$O + KI, ESI, 10 eV) = 720 (100, M$^+$ + OH$^-$), 368 (M$^+$ + 2 OH$^-$). IR (KBr) = 3443 (w), 3061 (w), 2957 (m), 2918 (m), 2856 (w), 2729 (w), 1725 (w), 1606 (s), 1592 (s), 1454 (w), 1466 (m), 1421 (m), 1402 (m), 1357 (w), 1278 (w), 1238 (s), 1214 (s), 1172 (s), 1154 (m), 1101 (w), 1030 (w), 979 (m), 874 (m), 846 (s), 818 (w), 798 (m), 744 (w), 724 (m), 663 (w), 586 (w), 562 (w), 515 (w). UV-Vis: 386 nm ($\epsilon = 65984$), 406 nm ($\epsilon = 65378$). quantum yield: $\Phi = 0.74 \pm 0.10$.

Crazy Output for these options:

yield: 17 mg yellow needles (0.04 mmol, 13%). mp. = 277 °C (DSC).
7. User Modules

\[ ^1H-NMR \ (600 \ MHz, \ CDCl_3) : \delta \ M^+ + H_2O = 2.01 \ (s, \ 24 \ H, \ H-5), \ 2.31 \ (s, \ 12 \ H, \ H-1), \ 6.72 - 6.74 \ (m, \ 2 \ H, \ H-11), \ 6.82 \ (s, \ 8 \ H, \ H-3), \ 7.05 - 7.07 \ (m, \ 2 \ H, \ H-12), \ 7.39 - 7.41 \ (m, \ 4 \ H, \ H-9), \ 7.48 - 7.49 \ (m, \ 4 \ H, \ H-8). \]

\[ ^13C-NMR \ (150 \ MHz, \ CDCl_3) : \delta \ M^+ + H_2O = 21.2 \ (+, \ 4 \ C, \ C-1), \ 23.4 \ (+, \ 8 \ C, \ C-5), \ 126.0 \ (+, \ 4 \ C, \ C-9), \ 128.2 \ (+, \ 8 \ C, \ C-3), \ 130.8 \ (+, \ 2 \ C, \ C-12), \ 133.6 \ (+, \ 2 \ C, \ C-11), \ 137.0 \ (+, \ 4 \ C, \ C-8), \ 138.6 \ (q, \ 4 \ C, \ C-2), \ 140.6 \ (q, \ 2 \ C, \ C-10), \ 140.8 \ (q, \ 8 \ C, \ C-4), \ 141.8 \ (q, \ 4 \ C, \ C-6), \ 145.6 \ (q, \ 2 \ C, \ C-7). \]

\[ MS (DCP, EI, 60 \ eV) = 703 \ (2, \ M^+), \ 582 \ (1), \ 462 \ (1), \ 249 \ (13), \ 120 \ (41), \ 105 \ (100). \]

\[ MS (MeOH + H_2O + KI, ESI, 10 \ eV) = 720 \ (100, \ M^+ + \ OH^-), \ 368 \ (M^+ + 2 \ OH^-). \]

IR (KBr) = 3443 (w), 3061 (w), 2957 (m), 2918 (m), 2856 (w), 2729 (w), 1725 (w), 1606 (s), 1592 (s), 1545 (w), 1446 (m), 1401 (m), 1357 (w), 1287 (w), 1238 (s), 1214 (s), 1172 (s), 1154 (m), 1101 (w), 1030 (w), 979 (m), 874 (m), 846 (s), 818 (w), 798 (m), 744 (w), 724 (m), 663 (w), 586 (w), 562 (w), 515 (w).

UV-Vis: 386 nm (\( \varepsilon = 65 \ 984 \)), 406 nm (\( \varepsilon = 65 \ 378 \)).

quantum yield: \( \Phi = 0.74 \pm 0.10 \).

7.4. The thermodynamics Module

The thermodynamics module loads the siunitx package [Wri18].

7.4.1. The \texttt{\{}state\texttt{}} Macro

\begin{verbatim}
\state[(options)]{(symbol)}
\end{verbatim}

Typeset a state variable.

This macro can be used to write the thermodynamic state variables.

\begin{verbatim}
\state(A) \ state[supercript-left=f]{G}, \ state[supercript-right=\ch\{Na]\}{E}, \ state[supercript-right=\SI{1000}{\celsius}]{H}
\end{verbatim}

\( \Delta A^*, \Delta G^*, \Delta E_{Na}^*, \Delta H^{1000^\circ C} \)

These options are available:

\begin{verbatim}
thermodynamics » \texttt{pre} = \{(text)\} \hspace{1cm} \text{Default: \texttt{\{}change\texttt{}}state\texttt{\}}
\end{verbatim}

Code inserted before the variable. Inserted in text mode.

\begin{verbatim}
thermodynamics » \texttt{post} = \{(text)\} \hspace{1cm} \text{(initially empty)}
\end{verbatim}

Code inserted after the variable. Inserted in text mode.

\begin{verbatim}
thermodynamics » \texttt{supercript-left} = \{(text)\} \hspace{1cm} \text{(initially empty)}
\end{verbatim}

The left superscript. Inserted in text mode.

\begin{verbatim}
thermodynamics » \texttt{supercript-right} = \{(text)\} \hspace{1cm} \text{Default: \texttt{\{}standard\texttt{}}state\texttt{\}}
\end{verbatim}

The right superscript. Inserted in text mode.
7. User Modules

thermodynamics » superscript = {⟨text⟩}  
An alias of superscript-right.

thermodynamics » subscript-left = {⟨text⟩}  
The left subscript. Inserted in text mode.  
(initially empty)

thermodynamics » subscript-right = {⟨text⟩}  
The right subscript. Inserted in text mode.  
(initially empty)

thermodynamics » subscript = {⟨text⟩}  
An alias of subscript-left.

7.4.2. Thermodynamic Variables

The thermodynamics module provides a few commands for specific thermodynamic variables:

\enthalpy*[{⟨options⟩}]{⟨subscript⟩}{⟨value⟩}  
Typeset the amount of enthalpy.

\entropy*[{⟨options⟩}]{⟨subscript⟩}{⟨value⟩}  
Typeset the amount of entropy.

\gibbs*[{⟨options⟩}]{⟨subscript⟩}{⟨value⟩}  
Typeset the amount of Gibbs enthalpy.

Their usage is pretty much self-explaining:

\begin{verbatim}
1. \enthalpy{123} \par \Delta H^* = 123 \text{kJ mol}^{-1}
2. \entropy{123} \par S^* = 123 \text{J K}^{-1} \text{mol}^{-1}
3. \gibbs{123} \par \Delta G^* = 123 \text{kJ mol}^{-1}
\end{verbatim}

The argument ⟨⟨subscript⟩⟩ adds a subscript for specification, * hides number and unit:

\begin{verbatim}
1. \enthalpy{(r)}{123} \par \Delta_{r} H^* = 123 \text{kJ mol}^{-1}
2. \enthalpy*{(123)} \par \Delta H^*
\end{verbatim}

thermodynamics » pre = {⟨text⟩}  
Code inserted before the variable. Inserted in text mode.  
Default: \changestate

thermodynamics » post = {⟨text⟩}  
Code inserted after the variable. Inserted in text mode.  
(initially empty)

thermodynamics » superscript-left = {⟨text⟩}  
The left superscript. Inserted in text mode.  
(initially empty)

thermodynamics » superscript-right = {⟨text⟩}  
The right superscript. Inserted in text mode.  
Default: \standardstate

thermodynamics » superscript = {⟨text⟩}  
An alias of superscript-right.

thermodynamics » subscript-left = {⟨text⟩}  
The left subscript. Inserted in text mode.  
(initially empty)
thermodynamics \[\textbf{subscript-right} = \{\langle text\rangle\}\]

The right subscript. Inserted in text mode.

thermodynamics \[\textbf{subscript} = \{\langle text\rangle\}\]

An alias of \textbf{subscript-left}.

thermodynamics \[\textbf{subscript-pos} = \text{left}|\text{right}\]

Default: \text{left}

Determine whether the subscript given in \((\langle \text{subscript} \rangle)\) is placed to the left or the right of the variable.

thermodynamics \[\textbf{symbol} = \{\langle symbol\rangle\}\]

(initially empty)

The symbol of the variable. Inserted in math mode.

thermodynamics \[\textbf{unit} = \{\langle unit\rangle\}\]

(initially empty)

A valid \texttt{siunitx} unit.

The default values depend on the command.

\[
\begin{align*}
\text{enthalpy}[\text{unit=\texttt{kilo\,joule}]\{-285} \par & \Delta H^\ast = -285 \text{ kJ} \\
\text{gibbs}[\text{pre=}\{0} \par & \text{G}^\ast = 0 \text{ kJ mol}^{-1} \\
\text{entropy}[\text{pre=}\{\Delta,\text{superscript}\} = \{56.7} & \Delta S = 56.7 \text{ J K}^{-1} \text{ mol}^{-1}
\end{align*}
\]

The unit is set corresponding to the rules of \texttt{siunitx} and depends on its settings:

\[
\begin{align*}
\text{enthalpy}\{-1234.56e3} \par & \Delta H^\ast = -1234.56 \times 10^3 \text{ kJ mol}^{-1} \\
\text{sisetup}\{ & \text{per-mode=}\text{symbol}, \\
\text{exponent-product=}\text{\cdot,} & \Delta H^\ast = -1234.56 \cdot 10^3 \text{ kJ/mol} \\
\text{output-decimal-marker=}\{,\}, & \text{group-four-digits=}\text{true} \\
\} & \textbf{enthalpy}\{-1234.56e3}
\end{align*}
\]

7.4.3. \textit{Create New Variables or Redefine Existing Ones}

\texttt{\textbackslash NewChemState}\{(cs)\}\{(options)\}

Define new state commands like \texttt{enthalpy}. Gives an error if \texttt{(cs)} already exists.

\texttt{\textbackslash RenewChemState}\{(cs)\}\{(options)\}

Redefine existing state commands.

\texttt{\textbackslash DeclareChemState}\{(cs)\}\{(options)\}

Like \texttt{\textbackslash NewChemState} but gives now error if \texttt{(cs)} already exists.

\texttt{\textbackslash ProvideChemState}\{(cs)\}\{(options)\}

Define new state commands like \texttt{enthalpy}. Defines \texttt{(cs)} only if it is not defined, yet.

The argument \texttt{(options)} is a comma separated list of key/value options:

thermodynamics \[\textbf{pre} = \{\langle text\rangle\}\]

Code inserted before the variable. Inserted in text mode.

Default: \texttt{\textbackslash changestate}
7. User Modules

`thermodynamics` \post = \{(text)\} 
\begin{itemize}
\item Code inserted after the variable. Inserted in text mode.
\end{itemize}

`thermodynamics` \superscript-left = \{(text)\} 
\begin{itemize}
\item The left superscript. Inserted in text mode.
\end{itemize}

`thermodynamics` \superscript-right = \{(text)\} 
\begin{itemize}
\item Default: `\standardstate`
\item The right superscript.
\end{itemize}

`thermodynamics` \superscript = \{(text)\} 
\begin{itemize}
\item An alias of `\superscript-right`.
\end{itemize}

`thermodynamics` \subscript-left = \{(text)\} 
\begin{itemize}
\item The left subscript. Inserted in text mode.
\end{itemize}

`thermodynamics` \subscript-right = \{(text)\} 
\begin{itemize}
\item The right subscript. Inserted in text mode.
\end{itemize}

`thermodynamics` \subscript = \{(text)\} 
\begin{itemize}
\item An alias of `\subscript-left`.
\end{itemize}

`thermodynamics` \subscript-pos = left\,|\,right 
\begin{itemize}
\item Default: `left`
\item Determines wether the subscript given in `(\subscript)` is placed to the left or the right of the variable.
\end{itemize}

`thermodynamics` \symbol = \{(symbol)\} 
\begin{itemize}
\item The symbol of the variable.
\end{itemize}

`thermodynamics` \unit = \{(unit)\} 
\begin{itemize}
\item A valid `siunitx` unit.
\end{itemize}

\begin{verbatim}
\NewChemState\Helmholtz{ symbol=A , unit=\kilo\joule\per\mole }
\NewChemState\ElPot{ symbol=E , subscript-pos=right , superscript= , unit=\volt }
\Helmholtz{123.4} \par
\ElPot{-1.1} \par
\ElPot[\superscript=0]{\ch{\Sn}||\ch{\Sn^{2+}}||\ch{\Pb^{2+}}||\ch{\Pb}}{0.01} \par
\RenewChemState\enthalpy{ symbol=h , unit=\joule } \par
\enthalpy(f){12.5}
\end{verbatim}

\begin{itemize}
\item $\Delta A^\circ = 123.4 \text{ kJ mol}^{-1}$
\item $\Delta E = -1.1 \text{ V}$
\item $\Delta E^\circ_{\Sn^{2+}|\Pb^{2+}|\Pb} = 0.01 \text{ V}$
\item $\Delta h^\circ = 12.5 \text{ J}$
\end{itemize}

The existing commands have been defined like this:

\begin{verbatim}
\NewChemState\enthalpy{ symbol = H , unit = \kilo\joule\per\mole }
\NewChemState\entropy { symbol = S , unit = \joule\per\kelvin\per\mole , pre = }
\NewChemState\gibbs { symbol = G , unit = \kilo\joule\per\mole }
\end{verbatim}
7. User Modules

So – for following thermodynamic conventions – one could define a molar and an absolute variable:

\begin{verbatim}
\RenewChemState\enthalpy{symbol=h, superscript=, unit=kilo/joule/per/mole}\% molar
\NewChemState\Enthalpy{symbol=H, superscript=, unit=kilo/joule}\% absolute
\enthalpy{-12.3} \Enthalpy{-12.3}
\end{verbatim}

\[ \Delta h = -12.3 \text{ kJ mol}^{-1} \quad \Delta H = -12.3 \text{ kJ} \]

7.5. The units Module

The units module loads the siunitx package [Wri18].

In chemistry some non-SI units are very common. siunitx provides the command

\texttt{\DeclareSIUnit{\hspace{1em}}{\hspace{1em}}}

Define \hspace{1em} to be a valid unit command inside \hspace{1em} macros \texttt{\SI} and \texttt{\si} which represents \hspace{1em}.

to add arbitrary units. \texttt{CHEMMACROS} uses that command to provide some units. Like all siunitx units they’re only valid inside \texttt{\SI{\hspace{1em}}{\hspace{1em}}} and \texttt{\si{\hspace{1em}}}.

\texttt{\atmosphere}

atm

\texttt{\atm}

atm

\texttt{\calory}

cal

\texttt{\cal}

cal

\texttt{\cmc}

\texttt{\cm}

cm\textsuperscript{3}

The units \texttt{\cmc}, \texttt{\molar}, and \texttt{\Molar} are defined by the package \texttt{chemstyle} as well. \texttt{CHEMMACROS} only defines them, if \texttt{chemstyle} is not loaded.

\texttt{\molar}

mol dm\textsuperscript{3}

\texttt{\moLar}

mol L\textsuperscript{-1}

\texttt{\Molar}

M

\texttt{\MolMass}

g mol\textsuperscript{-1}

\texttt{\normal}

N
\Torr

By the way: \texttt{mmHg} mmHg already is defined by \texttt{siunitx}.

# 8. Internal Modules

## 8.1. The \texttt{tikz} Module

The \texttt{tikz} module loads the \texttt{tikz} package [Tan19] and the \texttt{TikZ} library \texttt{calc}.

### 8.1.1. For Users

The \texttt{tikz} module defines a few arrow tips:

\begin{itemize}
  \item \texttt{el}
    \begin{enumerate}
      \item An arrow tip: \texttt{\tikz\draw[-el](0,0)--(1,0)}; \quad \rightarrow
    \end{enumerate}
  \item \texttt{left el}
    \begin{enumerate}
      \item An arrow tip: \texttt{\tikz\draw[-left el](0,0)--(1,0)}; \quad \rightarrow
    \end{enumerate}
  \item \texttt{right el}
    \begin{enumerate}
      \item An arrow tip: \texttt{\tikz\draw[-right el](0,0)--(1,0)}; \quad \rightarrow
    \end{enumerate}
\end{itemize}

The \texttt{tikz} module also loads the libraries \texttt{calc} and \texttt{decorations.pathmorphing}. It uses those libraries for defining a new decoration \texttt{wave}.

\begin{tikzpicture}
\draw[decorate,decoration=wave](0,0) -- (2,0);
\end{tikzpicture}

### 8.1.2. For Module Writers

The \texttt{tikz} module provides some macros for common \texttt{TikZ} functions. This allows to use \texttt{expl3}'s powerful function variants for expansion control.

\texttt{c.chemmacros_other_colon_tl}

A constant tokenlist which contains a colon with category code 12 (other). This is useful since \texttt{TikZ} sometimes expects an other colon and in an \texttt{expl3} programming environment: has category code 11 (letter).

\texttt{chemmacros_tikz_picture:nn}{\{\texttt{options}\}}{\{\texttt{code}\}}

Defined as \texttt{\tikzpicture}{\{\texttt{#1}\}} #2 \texttt{\endtikzpicture}.

\texttt{chemmacros_tikz:nn}{\{\texttt{options}\}}{\{\texttt{code}\}}

Defined as \texttt{\tikz}{\{\texttt{#1}\}}{\{\texttt{#2}\}}.

\texttt{chemmacros_tikz_draw:n}{\{\texttt{options}\}}

Defined as \texttt{\draw}{\{\texttt{#1}\}}.

\texttt{chemmacros_tikz_node:n}{\{\texttt{options}\}}

Defined as \texttt{\node}{\{\texttt{#1}\}}.
8. Internal Modules

### Table 4: Predefined \texttt{xfrac} text instances.

<table>
<thead>
<tr>
<th>font family</th>
<th>text</th>
<th>superscript</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmr</td>
<td>$\frac{2}{3}$</td>
<td>$\frac{2}{3}$</td>
</tr>
<tr>
<td>lmr</td>
<td>$\frac{2}{3}$</td>
<td>$\frac{2}{3}$</td>
</tr>
<tr>
<td>LinuxLibertineT-TLF</td>
<td>$\frac{1}{3}$</td>
<td>$\frac{2}{3}$</td>
</tr>
<tr>
<td>LinuxLibertineT-T0sF</td>
<td>$\frac{1}{3}$</td>
<td>$\frac{2}{3}$</td>
</tr>
</tbody>
</table>

\texttt{\chemmacros_tikz_shade:n \{options\}}

Defined as \texttt{\shade[#1]}.

\texttt{\chemmacros_tikz_shadedraw:n \{options\}}

Defined as \texttt{\shadedraw[#1]}.

\texttt{\chemmacros_tikz_node_in_draw:n \{options\}}

Defined as \texttt{node[#1]}.

### 8.2. The \texttt{xfrac} Module

The \texttt{xfrac} module loads the package \texttt{xfrac} \cite{L3Pb}. For the following explanations it will be helpful if you know about said package and how it works first. This module is a support module that defines the macro

\texttt{\chemfrac[\{type\}][\{numerator\}][\{denominator\}]

\texttt{(type)} can either be \texttt{text} or \texttt{superscript}.

This macro calls a certain instance of the \texttt{xfrac} text template, depending on the option \texttt{(type)} and the current font family. If used \texttt{\chemfrac} looks if an instance

\texttt{chemmacros-frac-\f@family-\{type\}}

exists. If yes this instance is used, if no the instance \texttt{chemmacros-frac-default-\{type\}} is used. The default instances are the same as the ones for \texttt{cmr}.

The \texttt{xfrac} module defines instances some font families, they are listed and demonstrated in table 4. The \texttt{superscript} type fractions look larger than the \texttt{text} types. The reason is that the \texttt{superscript} types are typically used with a smaller font size. Let’s take a look at an example where both instances are used:

\begin{verbatim}
1 \chemsetup[redox]{pos=top}
2 \code{superscript}:
3 \ch{"\ox{1/3,I}" {}3+}
4
5 \chemsetup[redox]{pos=side}
6 \code{text}:
7 \ox{1/3,I3+}
8 \huge
9 \chemsetup[redox]{pos=top}
10 \code{superscript}:
11 \ch{"\ox{1/3,I}" {}3+}
12
13 \chemsetup[redox]{pos=side}
14 \code{text}:
15 \ox{1/3,I3+}
\end{verbatim}

superscript: \(I_3^+\)

text: \(I_3^+(\frac{1}{3})\)
If you define instances for other families please feel free to submit them to me (see section A.2 on the next page) so they can be added to the xfrac module.

**Part IV. Appendix**

**A. Own Modules**

**A.1. How To**

If you have additional functionality which you think might be useful as a `chemmacros` module then you can easily write one yourself. The module must be a file in a path where \TeX\ can find it following a certain naming scheme. The file for a module `foo` must be named `chemmacros.module.foo.code.tex`.

```
\ChemModule*{foo}{2015/07/14 description of foo}
```

This registers module `foo` which means `chemmacros` will accept this file as a valid module. Since `chemmacros` is written using expl3 \ChemModule starts an expl3 programming environment. If you don’t want that but rather want to write your module using traditional \LaTeX methods then use the starred variant:

```
\ChemModule*{foo}{2015/07/14 description of foo}
```

In both variants `@` has category code 11 (letter). Since new modules very likely might rely on code provided first in a certain version of `chemmacros` you might want to make sure that your module only is loaded when the compatibility mode is high enough to provide the features you want:

```
\ChemModule{foo}{2015/10/14 description of foo}[5.2]
```

You should be aware that your module will not be loaded with `\usechemmodule{all}`! The pseudo-module `all` contains a manually maintained list of the modules that are loaded by it.

If you decide to write your module `foo` using expl3 and add options you want to be able to set using `\chemsetup[foo]{{options}}` please make sure you define (and set) them with the following macros:

```
\chemmacros_define_keys:nn {module} {key definitions}
```

Define \LaTeX keys options for the module `{module}`. This is a wrapper for `\keys_define:nn {chemmacros/}(module) {key definitions}`).
B. Suggestions, Bug Reports, Support

\chemmacros_set_keys:nn \{module\} \{input\}

Sets keys options for the module \{module\}. This is a wrapper for \keys_set:nn \{chemmacros/\{module\}\} \{\{input\}\}.

Also (especially if you consider submitting the module, see section A.2) please follow the expl3 naming conventions for variables and functions, i.e., use chemmacros as expl3 module name:

\tl_new:N \l__chemmacros_my_internal_variable_tl
\tl_new:N \l_chemmacros_my_public_variable_tl
\cs_new:Npn \__chemmacros_my_internal_function:n #1 { ... }
\cs_new:protected:Npn \chemmacros_my_public_function:n #1 { ... }
\NewDocumentCommand \publicfunction {m} { \chemmacros_my_public_function:n {#1} }

You will find more details on the naming conventions in interface3.pdf which most likely is available on your system:

~ $ \texttt{texdoc\hspace{1em}interface3}$

If you haven’t read section 6.1 starting on page 24 about the base module, yet, please have a look. There some macros for module writers are described. Also other modules define macros for module writers which may be useful.

A.2. Submitting a Module

If you have written a module and feel it might be useful for other users please feel free to contact me and submit the module. I will surely take a look at both functionality and code and if I feel that it adds value to \texttt{CHEMMACROS} I will add it to the package. Requirement for this is that the module is licensed with the \LaTeX{} Project Public License (v1.3 or later) and that I take over maintenance (according to the “maintainer” status of the LPPL).

Please do \textit{not} submit your module via pull request but send me the files directly. In the best case you also have a short piece of documentation.

B. Suggestions, Bug Reports, Support

Support If you need support or help with anything regarding \texttt{CHEMMACROS} please use the usual support forums

- http://www.golatex.de/ or
- http://texwelt.de/wissen/ if you speak German,
- http://www.latex-community.org/forum/ or
- http://tex.stackexchange.com/ if you speak English

or go the dedicated support forum


where you can be sure that I will see the question.
C. References

Suggestions If you have any suggestions on how chemmacros could be improved, adding missing features etc., please feel free to contact me via contact@mychemistry.eu.

Bug reports If you find any bugs, i.e., errors (something not working as described, conflicts with other packages,...) then please go to https://github.com/cgnieder/chemmacros/issues/ and open a new issue describing the error including a minimal working example.

C. References

  url: https://www.ctan.org/pkg/relsize.
  babel. version 3.33, July 19, 2019 (or newer).
  url: https://www.ctan.org/pkg/babel/.
  polyglossia. version 1.44, Apr. 4, 2019 (or newer).
  url: https://www.ctan.org/pkg/polyglossia/.
  bn. version 1.2d, July 24, 2019 (or newer). url: https://www.ctan.org/pkg/bm/.
[ Coh+08 ] E. Richard Cohan et al.
  “Quantities, Symbols and Units in Physical Chemistry”, IUPAC Green Book.
[ Fuj13 ] Shinsaku Fujita. X\textsc{m}t\textsc{x}. version 5.06, 2013 (or newer).
  url: https://www.ctan.org/pkg/xymtex/.
[ Hen18 ] Martin Hensel. mhchem. version 4.08, June 22, 2018 (or newer).
  url: https://www.ctan.org/pkg/mhchem/.
  url: https://www.ctan.org/pkg/koma-script/.
[ L3Pa ] The L\textsc{a}t\textsc{e}X3 Project Team. \LaTeX3 kernel. Mar. 6, 2020 (or newer).
  url: https://www.ctan.org/pkg/l3kernel/.
[ L3Pb ] The L\textsc{a}t\textsc{e}X3 Project Team. \LaTeX3 packages. Mar. 6, 2020 (or newer).
  url: https://www.ctan.org/pkg/l3packages/.
[ Leh19 ] Philipp Lehman, current maintainer: Joseph Wright.
  etoolbox. version 2.5h, Sept. 21, 2019 (or newer).
  url: https://www.ctan.org/pkg/etoolbox/.
[ MRW19 ] Lars Madsen, Will Robertson, and Joseph Wright.
  mathtools. version 1.22, July 31, 2019 (or newer).
  url: https://www.ctan.org/pkg/mh/.
[ MS00 ] Frank Mittelbach and Rainer Schöpf.
  amstext. version 2.01, June 29, 2000 (or newer).
  url: https://www.ctan.org/pkg/amstext/.
[ Nie15 ] Clemens Niederberger. elements. version 0.1, June 14, 2015 (or newer).
  url: https://www.ctan.org/pkg/elements/.
C. References

[Nie16a] Clemens Niederberger. chemgreek. version 1.1, Dec. 20, 2016 (or newer).
url: https://www.ctan.org/pkg/chemgreek/.

[Nie16b] Clemens Niederberger. chemnum. version 1.2, Apr. 14, 2016 (or newer).
url: https://www.ctan.org/pkg/chemnum/.

[Nie19] Clemens Niederberger. chemformula. version 4.15f, Sept. 23, 2019 (or newer).
url: https://www.ctan.org/pkg/chemformula/.

url: https://www.ctan.org/pkg/translations/.

url: https://www.ctan.org/pkg/bpchem/.


url: https://www.ctan.org/pkg/pgf/.

[Tel19] Christian Tellechea. chemfig. version 1.41, May 21, 2019 (or newer).
url: https://www.ctan.org/pkg/chemfig/.

url: https://www.ctan.org/pkg/chemstyle/.

[Wri18] Joseph Wright. siunitx. version 2.7s, May 17, 2018 (or newer).
url: https://www.ctan.org/pkg/siunitx/.
D. Index

Symbols

\(\cdot\) (symbol) .................................................. 13
\{(symbol) .................................................. 13
\} (symbol) .................................................. 13
\_ (symbol) .................................................. 13f.
\_\_ (symbol) ............................................ 14
\{ (symbol) .................................................. 13
\} (symbol) .................................................. 13
\$ .................................................. 53:55
\$ (symbol) .................................................. 13
\{ (symbol) .................................................. 13f.
\} (symbol) .................................................. 13

A

\% .................................................. 15
\%abinitio .................................................. 20
\%AddRxnDesc .................................................. 46
\%align .................................................. 47
\%alignat (environment) .................................................. 44
\%all (module) .................................................. 34:65
\%alt .................................................. 40
\%alternating .................................................. 40
\%amsmath (package) .................................................. 41
\%amstext (package) .................................................. 24:26:35
\%angle .................................................. 36:38
\%anti .................................................. 17:19
\%aq .................................................. 22f.
\%aqi .................................................. 23
\%Arseneau, Donald .................................................. 46
\%atm .................................................. 62
\%atmosphere .................................................. 62
\%atom-number-cs .................................................. 55
\%atom-number-space .................................................. 55
\%atoms .................................................. 36

B

\%b .................................................. 15
\%ba .................................................. 20 ff.
\%babel (package) .................................................. 24:30
\%back-atom .................................................. 36
\%base (module) .................................................. 24:26:66
\%before-tag .................................................. 43
\%Bezos, Javier .................................................. 40
\%blend .................................................. 40
\%block .................................................. 40
\%bm (package) .................................................. 49
\%bond .................................................. 38f.
\%boolean-option .................................................. 4
\%bpchem (package) .................................................. 13
\%braams, Johannes .................................................. 30
\%branch .................................................. 40
\%break-space .................................................. 14
\%bridge .................................................. 18
\%bridge-number .................................................. 18

C

\%chemmacros_date.tl (expl3) .................................................. 7
\%chemmacros_info.tl (expl3) .................................................. 7
\%chemmacros_other_colon.tl (expl3) .................................................. 63
\c_chemmacros_version_major.number.tl (expl3) .................................................. 7
\c_chemmacros_version_minor.number.tl (expl3) .................................................. 7
\c_chemmacros_version_number.tl (expl3) .................................................. 7
\c_chemmacros_version_subrelease.tl (expl3) .................................................. 7
\c_chemmacros_version.tl (expl3) .................................................. 7
\cal .................................................. 62
\ch .................................................. 62
\chall .................................................. 23
\challissle, David .................................................. 24
\cd .................................................. 27 ff.
\ce .................................................. 41
\che .................................................. 5:11, 20, 22 ff., 27, 29, 41, 47–50, 56 ff., 61, 64
\changestate .................................................. 24:58 ff.
\charrette, François .................................................. 30
\charges (module) .................................................. 5, 10, 20, 26, 28
\chcmd .................................................. 5:21, 27 ff., 46, 52
\chemabove .................................................. 11
\chemalpha .................................................. 15
\chemalta .................................................. 15
\chembeta .................................................. 15
\chemclevereffect .................................................. 25
\chemcompatibility .................................................. 7
\chemcomparitybetween .................................................. 7
\chemcompatibilityfrom .................................................. 7
\chemcompatibilityto .................................................. 7
\chemdelta .................................................. 25
\chemfancyrefformula .................................................. 26
\chemfancyrefformula .................................................. 26
\chemfig (package) .................................................. 3:10 f., 26 ff., 49
\chemform .................................................. 28
\chemformulal .................................................. 27
\chemformula (module) .................................................. 10, 20, 22, 26 ff., 41
\chemformulal (package) .................................................. 5:21, 27 ff., 46, 52
\chemfrac .................................................. 47:64
\chemgamma .................................................. 15
\chemgreek (package) .................................................. 8:29
\chemist (package) .................................................. 27 ff.
\chemmacros_add_cleveref_support:nnnn .................................................. 25
\chemmacros_add_fancyrefformula .................................................. 25
\chemmacros_add_fancyrefformula .................................................. 26
\chemmacros_after_module:nn .................................................. 26
\chemmacros_allow_brea .................................................. 25
\chemmacros_before_module:nn .................................................. 26
\chemmacros_bold:n .................................................. 25
\chemmacros_cheformula:n .................................................. 28
\chemmacros_declare_translation:nn .................................................. 33
\chemmacros_declare_translation:nn .................................................. 33
\chemmacros_define_keys:nn .................................................. 65
\chemmacros_if_bold: .................................................. 25
\chemmacros_if_class_loaded:n .................................................. 25
\chemmacros_if_compatibility:nn .................................................. 7
\chemmacros_if_is_int:n .................................................. 25
\chemmacros_if_loaded:nn .................................................. 24
\chemmacros_if_module_exist:n .................................................. 26
\chemmacros_if_module_loaded:n .................................................. 26
\chemmacros_if_package_loaded:n .................................................. 24
\chemmacros_load_module:n .................................................. 26
\chemmacros_load_modules:n .................................................. 26
\chemmacros_maths:n .................................................. 25