siunitx – A comprehensive (SI) units package*

Joseph Wright†

Released 2018-07-21

Contents

*This file describes v3.0.0-alpha, last revised 2018-07-21.

†E-mail: joseph.wright@morningstar2.co.uk

Abstract

Physical quantities have both numbers and units, and each physical quantity should be expressed as the product of a number and a unit. Typesetting physical quantities requires care to ensure that the combined mathematical meaning of the number–unit combination is clear. In particular, the SI units system lays down a consistent set of units with rules on how these are to be used. However, different countries and publishers have differing conventions on the exact appearance of numbers (and units). The siunitx package provides a set of tools for authors to typeset quantities in a consistent way. The package has an extended set of configuration options which make it possible to follow varying typographic conventions with the same input syntax. The package includes automated processing of numbers and units, and the ability to control tabular alignment of numbers.

1 Introduction

The correct application of units of measurement is very important in technical applications. For this reason, carefully-crafted definitions of a coherent units system have been laid down by the *Conférence Génrale des Poids et Mesures* (CGPM): this has resulted in the *Système International d'Unités* (SI). At the same time, typographic conventions for correctly displaying both numbers and units exist to ensure that no loss of meaning occurs in printed matter.

The siunitx package aims to provide a unified method for LaTeX users to typeset numbers and units correctly and easily. The design philosophy of siunitx is to follow the agreed rules by default, but to allow variation through option settings. In this way, users can use siunitx to follow the requirements of publishers, co-authors, universities, *etc.* without needing to alter the input at all.

2 Using the siunitx package

2.1 Numbers

`\num`

`\num[⟨options⟩]{⟨number⟩}`

Numbers are automatically formatted by the `\num` macro. This takes one optional argument, ⟨*options*⟩, and one mandatory one, ⟨*number*⟩. The contents of ⟨*number*⟩ are automatically formatted. The formatter removes both "soft" (␣) and "hard" spaces (`\,` and `~`), automatically identifies exponents (as standard marked using `e`, `E`, `d` or `D`) and adds the appropriate spacing of large numbers. With the standard settings a leading zero is added before a decimal marker, if needed: both `.` and `,` are recognised as decimal markers.

123	`\num{123}      \\`
1 234	`\num{1234}     \\`
12 345	`\num{12345}    \\`
0.123	`\num{0.123}    \\`
0.123 4	`\num{0,1234}   \\`
0.123 45	`\num{.12345}   \\`
3.45×10^{-4}	`\num{3.45d-4}  \\`
-10^{10}	`\num{-e10}`

Note that numbers are parsed before typesetting, which does have a performance overhead (only obvious with very large amounts of numerical input). The parser understands a range of input syntaxes, as demonstrated above.

2.2 Angles

\ang

\ang[⟨*options*⟩]{⟨*angle*⟩}

Angles can be typeset using the \ang command. The ⟨*angle*⟩ can be given either as a decimal number or as a semi-colon separated list of degrees, minutes and seconds, which is called "arc format" in this document. The numbers which make up an angle are processed using the same system as other numbers.

$10°$	`\ang{10}    \\`
$12.3°$	`\ang{12.3}  \\`
$4.5°$	`\ang{4,5}   \\`
$1°2'3''$	`\ang{1;2;3} \\`
$1''$	`\ang{;;1}   \\`
$10°$	`\ang{+10;;} \\`
$-0°1'$	`\ang{-0;1;}`

2.3 Units

\unit

\unit[⟨*options*⟩]{⟨*unit*⟩}

The symbol for a unit can be typeset using the \unit macro: this provides full control over output format for the unit. Like the \num macro, \unit takes one optional and one mandatory argument. The unit formatting system can accept two types of input. When the ⟨*unit*⟩ contains literal items (for example letters or numbers) then siunitx converts . and ~ into inter-unit product and correctly positions sub- and superscripts specified using _ and ^. The formatting methods will work with both math and text mode.

$\mathrm{kg\,m/s^2}$	`\unit{kg.m/s^2} \\`
$\mathrm{g_{polymer}\,mol_{cat}\,s^{-1}}$	`\unit{g_{polymer}~mol_{cat}.s^{-1}}`

The second operation mode for the \unit macro is an "interpreted" system, Here, each unit, SI multiple prefix and power is given a macro name. These are entered in a method very similar to the reading of the unit name in English.

```
\unit{\kilo\gram\metre\per\square\second} \\
\unit{\gram\per\cubic\centi\metre}        \\
\unit{\square\volt\cubic\lumen\per\farad} \\
\unit{\metre\squared\per\gray\cubic\lux}  \\
\unit{\henry\second}
```

$\mathrm{kg\,m\,s^{-2}}$
$\mathrm{g\,cm^{-3}}$
$\mathrm{V^2\,lm^3\,F^{-1}}$
$\mathrm{m^2\,Gy^{-1}\,lx^3}$
$\mathrm{H\,s}$

On its own, this is less convenient than the direct method, although it does use meaning rather than appearance for input. However, the package allows you to define new unit macros; a large number of pre-defined abbreviations are also supplied. More importantly, by defining macros for units, instead of literal input, new functionality is made available. By altering the settings used by the package, the same input can yield a variety of different output formats. For example, the `\per` macro can give reciprocal powers, slashes or be used to construct units as fractions.

`\qty`

`\qty[⟨options⟩]{⟨number⟩}{⟨unit⟩}`

Very often, numbers and units are given together. Formally, the value of a quantity is the product of the number and the unit, the space being regarded as a multiplication sign [5]. The `\qty` macro combines the functionality of `\num` and `\unit`, and makes this both possible and easy. The ⟨*number*⟩ and ⟨*unit*⟩ arguments work exactly like those for the `\num` and `\unit` macros, respectively.

```
\qty[mode = text]{1.23}{J.mol^{-1}.K^{-1}}          \\
\qty{.23e7}{\candela}                               \\
\qty[per-mode = symbol]{1.99}{\per\kilogram}        \\
\qty[per-mode = fraction]{1,345}{\coulomb\per\mole}
```

1.23 J mol^{-1} K^{-1}
0.23×10^{7} cd
1.99 1/kg
1.345 $\frac{\mathrm{C}}{\mathrm{mol}}$

It is possible to set up the unit macros to be available outside of the `\qty` and `\unit` functions. This is not the standard behaviour as there is the risk of name clashes (for example, `\bar` is used by other packages, and several packages define `\degree`). Full details of using "stand alone" units are found in **??**.

2.4 The unit macros

The package always defines the basic set of SI units with macro names. This includes the base SI units, the derived units with special names and the prefixes. A small number of powers are also given pre-defined names. Full details of units in the SI are available on-line [1].

`\meter` The seven base SI units are always defined (Table 1). In addition, the macro `\meter` is available as an alias for `\metre`, for users of US spellings. The full details of the base units are given in the SI Brochure [2].

The SI also lists a number of units which have special names and symbols [3]: these are listed in Table 2.

In addition to the official SI units, siunitx also provides macros for a number of units which are accepted for use in the SI although they are not SI units. Table 3 lists the "accepted" units [7]. Some units are fundamental physical quantities, and these are non-SI but can be used within the SI (Table 4, [8]). There are also a set of non-SI units which are used in certain defined circumstances (Table 5), although they are not necessarily officially sanctioned [9]. Finally, for specialist applications, a range of CGS units are

`\percent` sanctioned (Table 6) [10]. The command `\percent` is also provided for use in units: this is accepted with the SI as detailed in Section 5.3.7 of the Brochure [6].

`\deka` In addition to the units themselves, siunitx provides pre-defined macros for all of the

Table 1: SI base units.

Unit	Command	Symbol
ampere	`\ampere`	A
candela	`\candela`	cd
kelvin	`\kelvin`	K
kilogram	`\kilogram`	kg
metre	`\metre`	m
mole	`\mole`	mol
second	`\second`	s

Table 2: Coherent derived units in the SI with special names and symbols.

Unit	Command	Symbol	Unit	Command	Symbol
becquerel	`\becquerel`	Bq	newton	`\newton`	N
degree Celsius	`\degreeCelsius`	°C	ohm	`\ohm`	Ω
coulomb	`\coulomb`	C	pascal	`\pascal`	Pa
farad	`\farad`	F	radian	`\radian`	rad
gray	`\gray`	Gy	siemens	`\siemens`	S
hertz	`\hertz`	Hz	sievert	`\sievert`	Sv
henry	`\henry`	H	steradian	`\steradian`	sr
joule	`\joule`	J	tesla	`\tesla`	T
lumen	`\lumen`	lm	volt	`\volt`	V
katal	`\katal`	kat	watt	`\watt`	W
lux	`\lux`	lx	weber	`\weber`	Wb

Table 3: Non-SI units accepted for use with the International System of Units.

Unit	Command	Symbol
day	`\day`	d
degree	`\degree`	°
hectare	`\hectare`	ha
hour	`\hour`	h
litre	`\litre`	L
	`\liter`	L
minute (plane angle)	`\arcminute`	′
minute (time)	`\minute`	min
second (plane angle)	`\arcsecond`	″
tonne	`\tonne`	t

Table 4: Non-SI units whose values in SI units must be obtained experimentally.

Unit	Command	Symbol
astronomical unit	\astronomicalunit	au
atomic mass unit	\atomicmassunit	u
a.u. of action	\auaction	$\hbar$
a.u. of charge	\aucharge	e
a.u. of energy	\auenergy	E_h
a.u. of length	\aulength	a_0
a.u. of mass	\aumass	m_e
a.u. of time	\autime	$\hbar\,{E_\mathrm{h}}^{-1}$
bohr	\bohr	a_0
dalton	\dalton	Da
electronvolt	\electronvolt	eV
hartree	\hartree	E_h
n.u. of action	\nuaction	$\hbar$
n.u. of mass	\numass	m_e
n.u. of speed	\nuspeed	c_0
n.u. of time	\nutime	$m_\mathrm{e}\,{m_\mathrm{e}}^{-1}\,{c_0}^{-2}$

Table 5: Other non-SI units.

Unit	Command	Symbol
ångström	\angstrom	Å
bar	\bar	bar
barn	\barn	b
bel	\bel	B
decibel	\decibel	dB
knot	\knot	kn
millimetre of mercury	\millimetremercury	mmHg
nautical mile	\nauticalmile	M
neper	\neper	Np

Table 6: Non-SI units associated with the CGS.

Unit	Command	Symbol
dyne	\dyne	dyn
erg	\erg	erg
gal	\gal	Gal
gauss	\gauss	G
maxwell	\maxwell	Mx
oersted	\oersted	Oe
phot	\phot	ph
poise	\poise	P
stilb	\stilb	sb
stokes	\stokes	St

Table 7: SI prefixes.

Prefix	Command	Symbol	Power	Prefix	Command	Symbol	Power
yocto	`\yocto`	y	−24	deca	`\deca`	da	1
zepto	`\zepto`	z	−21	hecto	`\hecto`	h	2
atto	`\atto`	a	−18	kilo	`\kilo`	k	3
femto	`\femto`	f	−15	mega	`\mega`	M	6
pico	`\pico`	p	−12	giga	`\giga`	G	9
nano	`\nano`	n	−9	tera	`\tera`	T	12
micro	`\micro`	µ	−6	peta	`\peta`	P	15
milli	`\milli`	m	−3	exa	`\exa`	E	18
centi	`\centi`	c	−2	zetta	`\zetta`	Z	21
deci	`\deci`	d	−1	yotta	`\yotta`	Y	24

SI prefixes (Table 7) [4]. The spelling "`\deka`" is provided for US users as an alternative to `\deca`.

`\square` `\squared` `\cubic` `\cubed`

A small number of pre-defined powers are provided as macros. `\square` and `\cubic` are intended for use before units, with `\squared` and `\cubed` going after the unit.

Bq^{2}	`\unit{\square\becquerel} \\`
$\mathrm{J}^{2}\,\mathrm{lm}^{-1}$	`\unit{\joule\squared\per\lumen} \\`
$\mathrm{lx}^{3}\,\mathrm{V}\,\mathrm{T}^{3}$	`\unit{\cubic\lux\volt\tesla\cubed}`

`\tothe` `\raiseto`

Generic powers can be inserted on a one-off basis using the `\tothe` and `\raiseto` macros. These are the only macros for units which take an argument:

H^{5}	`\unit{\henry\tothe{5}} \\`
$\mathrm{rad}^{4.5}$	`\unit{\raiseto{4.5}\radian}`

`\per`

Reciprocal powers are indicated using the `\per` macro. This applies to the next unit only, unless the `sticky-per` option is turned on.

$\mathrm{J}\,\mathrm{mol}^{-1}\,\mathrm{K}^{-1}$	`\unit{\joule\per\mole\per\kelvin} \\`
$\mathrm{J}\,\mathrm{mol}^{-1}\,\mathrm{K}$	`\unit{\joule\per\mole\kelvin} \\`
H^{-5}	`\unit{\per\henry\tothe{5}} \\`
Bq^{-2}	`\unit{\per\square\becquerel}`

`\of`

As for generic powers, generic qualifiers are also available using the `\of` function:

```
\unit{\kilogram\of{metal}} \\
\qty[qualifier-mode = bracket]
  {0.1}{\milli\mole\of{cat}\per\kilogram\of{prod}}
```

$\mathrm{kg}_{\mathrm{metal}}$
$0.1\,\mathrm{mmol(cat)}\,\mathrm{kg(prod)}^{-1}$

2.5 Unit abbreviations

In addition to the "full" names, siunitx loads a set of abbreviated versions of the SI units (Table 2.5). The standard siunitx settings only create these abbreviations within the scope of the `\unit` and `\qty` functions, meaning that no clashes should occur (for example with the standard `\pm` symbol).

Unit	Abbreviation	Symbol
femtogram	`\fg`	fg
picogram	`\pg`	pg
nanogram	`\ng`	ng
microgram	`\ug`	µg
milligram	`\mg`	mg
gram	`\g`	g
kilogram	`\kg`	kg
atomic mass unit	`\amu`	u
picometre	`\pm`	pm
nanometre	`\nm`	nm
micrometre	`\um`	µm
millimetre	`\mm`	mm
centimetre	`\cm`	cm
decimetre	`\dm`	dm
metre	`\m`	m
kilometre	`\km`	km
attosecond	`\as`	as
femtosecond	`\fs`	fs
picosecond	`\ps`	ps
nanosecond	`\ns`	ns
microsecond	`\us`	µs
millisecond	`\ms`	ms
second	`\s`	s
femtomole	`\fmol`	fmol
picomole	`\pmol`	pmol
nanomole	`\nmol`	nmol
micromole	`\umol`	µmol
millimole	`\mmol`	mmol
mole	`\mol`	mol
kilomole	`\kmol`	kmol
picoampere	`\pA`	pA
nanoampere	`\nA`	nA
microampere	`\uA`	µA

Continued on next page

Continued from previous page

Unit	Abbreviation	Symbol
milliampere	`\mA`	mA
ampere	`\A`	A
kiloampere	`\kA`	kA
microlitre	`\ul`	µL
millilitre	`\ml`	mL
litre	`\l`	L
hectolitre	`\hl`	hL
microliter	`\uL`	µL
milliliter	`\mL`	mL
liter	`\L`	L
hectoliter	`\hL`	hL
millihertz	`\mHz`	mHz
hertz	`\Hz`	Hz
kilohertz	`\kHz`	kHz
megahertz	`\MHz`	MHz
gigahertz	`\GHz`	GHz
terahertz	`\THz`	THz
millinewton	`\mN`	mN
newton	`\N`	N
kilonewton	`\kN`	kN
meganewton	`\MN`	MN
pascal	`\Pa`	Pa
kilopascal	`\kPa`	kPa
megapacal	`\MPa`	MPa
gigapascal	`\GPa`	GPa
milliohm	`\mohm`	mΩ
kilohm	`\kohm`	kΩ
megohm	`\Mohm`	MΩ
picovolt	`\pV`	pV
nanovolt	`\nV`	nV
microvolt	`\uV`	µV
millivolt	`\mV`	mV
volt	`\V`	V
kilovolt	`\kV`	kV
watt	`\W`	W
microwatt	`\uW`	µW

Continued on next page

Continued from previous page

Unit	Abbreviation	Symbol
milliwatt	`\mW`	mW
kilowatt	`\kW`	kW
megawatt	`\MW`	MW
gigawatt	`\GW`	GW
joule	`\J`	J
microjoule	`\uJ`	µJ
millijoule	`\mJ`	mJ
kilojoule	`\kJ`	kJ
electronvolt	`\eV`	eV
millielectronvolt	`\meV`	meV
kiloelectronvolt	`\keV`	keV
megaelectronvolt	`\MeV`	MeV
gigaelectronvolt	`\GeV`	GeV
teraelectronvolt	`\TeV`	TeV
kilowatt hour	`\kWh`	kW h
farad	`\F`	F
femtofarad	`\fF`	fF
picofarad	`\pF`	pF
kelvin	`\K`	K
decibel	`\dB`	dB

2.6 Tabular material

Aligning numbers in tabular content is handled by a new column type, the `S` column. This new column type can align material using a number of different strategies, with the aim of flexibility of output without needing to alter the input. The method used as standard is to place the decimal marker in the number at the centre of the cell and to align the material appropriately (Table 8).

```
\begin{table}
  \caption{Standard behaviour of the \texttt{S} column type.}
  \label{tab:S:standard}
  \begin{tabular}{S}
  \toprule
    {Some Values} \\
  \midrule
       2.3456 \\
      34.2345 \\
      -6.7835 \\
      90.473  \\
    5642.5    \\
       1.2e3 \\
         e4  \\
  \bottomrule
  \end{tabular}
\end{table}
```

The `S` column will attempt to automatically detect material which should be placed before or after a number, and will maintain the alignment of the numerical data (Table 9).

Table 8: Standard behaviour of the `S` column type.

Some Values
2.345 6
34.234 5
−6.783 5
90.473
5 642.5
1.2×10^3
10^4

Table 9: Detection of surrounding material in an `S` column.

Some Values
12.34
975,31
44.268*a*

If the material could be mistaken for part of a number, it should be protected by braces. The use of `\color` in a table cell will also be detected and will override any general colour applied by siunitx.

```
\begin{table}
  \caption{Detection of surrounding material in an \texttt{S}
    column.}
  \label{tab:S:extras}
  \begin{tabular}{S[color = orange]}
  \toprule
    {Some Values} \\
  \midrule
    12.34 \\
    \color{purple} 975,31 \\
    44.268 \textsuperscript{\emph{a}} \\
  \bottomrule
  \end{tabular}
\end{table}
```

`\tablenum`

`\tablenum[⟨options⟩]{⟨number⟩}`

Within more complex tables, aligned numbers may be desirable within the argument of `\multicolumn` or `\multirow`.[1] The `\tablenum` function is available to achieve alignment in these situations: this is, in effect, a macro version of the `S` column (Table 10).

```
\begin{table}
  \caption{Controlling complex alignment with the \cs{tablenum}
    macro.}
  \label{tab:tablenum}
  \begin{tabular}{@{}lr@{}}
```

[1] Provided by the multirow package

Table 10: Controlling complex alignment with the `\tablenum` macro.

Heading	Heading
Info	More info
Info	More info
12.34	
333.556 7	
4 563.21	

Heading	Heading
88.999	aaa
	bbb
33.435	ccc
	ddd

```
    \toprule
    Heading & Heading \\
    \midrule
    Info & More info \\
    Info & More info \\
    \multicolumn{2}{c}{\tablenum[table-format = 4.4]{12,34}}    \\
    \multicolumn{2}{c}{\tablenum[table-format = 4.4]{333.5567}} \\
    \multicolumn{2}{c}{\tablenum[table-format = 4.4]{4563.21}}  \\
    \bottomrule
  \end{tabular}
  \hfil
  \begin{tabular}{@{}lr@{}}
    \toprule
    Heading & Heading \\
    \midrule
    \multirow{2}*{\tablenum{88,999}} & aaa \\
                                     & bbb \\
    \multirow{2}*{\tablenum{33,435}} & ccc \\
                                     & ddd \\
    \bottomrule
  \end{tabular}
\end{table}
```

3 Package control options

3.1 The key–value control system

The package uses a range of different key types:

`Choice` Takes a limited number of choices, which are described separately for each key.

`Integer` Requires a number as the argument.

`Length` Requires a length, either as a literal value such as `2.0cm`, or stored as a LaTeX length.

`Literal` A key which uses the value(s) given directly, either to check input or in output.

`Macro` Requires a macro, which may need a single argument.

`Math` Similar to a `literal` option, but the input is always used in math mode, irrespective of other siunitx settings. Thus to text-mode only input must be placed inside the argument of a `\text` macro.

Table 11: Print options.

Option name	Type	Default
`color`	Literal	⟨*none*⟩
`mode`	Choice	`math`
`number-color`	Literal	⟨*none*⟩
`number-mode`	Choice	`math`
`propagate-math-font`	Switch	`false`
`reset-math-version`	Switch	`true`
`reset-text-family`	Switch	`true`
`reset-text-series`	Switch	`true`
`reset-text-shape`	Switch	`true`
`text-family-to-math`	Switch	`false`
`text-weight-to-math`	Switch	`false`
`unit-color`	Literal	⟨*none*⟩
`unit-mode`	Choice	`math`

Meta These are options which actually apply a number of other options.

Switch These are on–off switches, and recognise `true` and `false`. Giving just the key name also turns the key on.

The tables of option names use these descriptions to indicate how the keys should be used.

3.2 Printing

The siunitx package can control the font used to print output independently of the surrounding material. Which aspects of the font follow those of the surroundings is influenced by a range of setting as detailed in Section 11.

`mode`
`number-mode`
`unit-mode`

The `mode` option determines whether siunitx uses math or text mode when printing output. The choices are `match`, `math`, `text`. The `match` setting means that printing uses the prevailing mode unchanged whereas `math` and `text` select the appropriate TeX mode. It is possible to have different fonts in math and text modes, which will highlight the difference. The font settings which apply are also different depending on the mode. As well as the overall setting, it is possible to apply mode to numbers and units separately using the `number-mode` and `unit-mode` options.

`reset-text-family`
`reset-text-series`
`reset-text-shape`

When printing in text mode, the options `reset-text-family`, `reset-text-series` and `reset-text-shape` apply. When these are active, siunitx resets the relevant font selection axis property when printing: the standard font setting is upright mid-weight roman (`\upshape \mdseries \rmfamily`).

1 234
1 234
1 234
1 234
1 234
1 234

```
\sisetup{mode = text}
{\itshape  \num{1234}}\\
{\bfseries \num{1234}}\\
{\sffamily \num{1234}}\\
\sisetup{
  reset-text-family = false ,
  reset-text-series = false ,
  reset-text-shape  = false
}
{\itshape  \num{1234}}\\
{\bfseries \num{1234}}\\
{\sffamily \num{1234}}\\
```

`propagate-math-font`
`reset-math-version`

In math mode, the font used by LaTeX is "invariant", and this is reflected in the options available. With the standard settings, in math mode printing uses the standard math font and version (weight). The option `propagate-math-font` may be used to apply the a prevailing math font to the printed material. The setting `reset-math-version` controls whether the math version is reset or not. Note that math version is typically used to set "bold math" but may also be used for other effects, for example all sanserif math.

kg
kg
kg
kg
kg
kg

```
{\boldmath \unit{\kilogram}}\\
{\sansmath $\unit{\kilogram}$}\\
{$\mathsf{\unit{\kilogram}}$}\\
\sisetup{
  propagate-math-font = true  ,
  reset-math-version  = false
}
{\boldmath \unit{\kilogram}}\\
{\sansmath $\unit{\kilogram}$}\\
{$\mathsf{\unit{\kilogram}}$}
```

`text-family-to-math`
`text-weight-to-math`

The options `text-family-to-math` and `text-family-to-math` can be used to match (as far as possible) math mode output to the surrounding text. These options work by detecting the current text settings and making the appropriate choice in math mode.

kg
kg
kg
kg

```
{\sffamily \unit{\kilogram}}\\
{\bfseries $\unit{\kilogram}$}\\
\sisetup{
  text-family-to-math = true  ,
  text-weight-to-math = true
}
{\sffamily \unit{\kilogram}}\\
{\bfseries $\unit{\kilogram}$}
```

`color`
`number-color`
`unit-color`

The color of printed output can be set using the `color` option. When no color is given, printing follows the surrounding text. In contrast, when a specific color is given, it is used irrespective of the surroundings. As with `mode`, the `color` setting may also be applied to numbers and units independently.

Table 12: Options for number parsing.

Option name	Type	Default
`evaluate-expression`	Switch	`false`
`expression`	Literal	`#1`
`input-close-uncertainty`	Literal	`)`
`input-comparators`	Literal	`<=>\approx\ge\geq \gg\le\leq\ll \sim`
`input-decimal-markers`	Literal	`.,`
`input-digits`	Literal	`0123456789`
`input-exponent-markers`	Literal	`dDeE`
`input-ignore`	Literal	⟨*none*⟩
`input-open-uncertainty`	Literal	`(`
`input-signs`	Literal	`+-\pm\mp`
`input-uncertainty-signs`	Literal	`\pm`
`parse-numbers`	Switch	`true`

Some text
4 kg
More text
4 kg
Still red here!

```
\color{red}%
Some text \\
\qty{4}{\kilogram} \\
More text \\
\qty[color = blue]{4}{\kilogram} \\
Still red here!
```

3.3 Parsing numbers

The package uses a sophisticated parsing system to understand numbers. This allows siunitx to carry out a range of formatting, as described later. All of the input options take lists of literal tokens, and are summarised in Table 12.

`input-digits`
`input-decimal-markers`
`input-signs`
`input-exponent-markers`

The basic parts of a number are the digits, any sign and a separator between the integer and decimal parts. These are stored in the input options `input-digits`, `input-decimal-markers` and `input-signs`, respectively. More than one input decimal marker can be used: it will be converted by the package to the appropriate output marker. Numbers which include an exponent part also require a marker for the exponent: this again is taken from the range of tokens in the `input-exponent-markers` option.

`input-ignore`

Tokens given in the `input-ignore` list are totally passed over by siunitx: they will be removed from the input with no further processing.

`input-comparators`

In addition to signs, siunitx can recognise comparators, such as `<`. The package will automatically carry out conversions for `<<`, `>>`, `<=` and `>=` to `\ll`, `\gg`, `\le` and `\ge`, respectively.

<10
$\gg 5$ m
≤ 0.12

```
\num{< 10} \\
\qty{>> 5}{\metre} \\
\num{\le 0.12}
```

`input-open-uncertainty`
`input-close-uncertainty`
`input-uncertainty-signs`

In some fields, it is common to give the uncertainty in a number in brackets after the main part of the number, for example "1.234(5)". The opening and closing symbols used for this type of input are set as `input-open-uncertainty` and `input-close-uncertainty`. Alternatively, the uncertainty may be given as a separate part following a sign. Which signs are valid for this operation is determined by the input-uncertainty-signs option. As with other signs, the combination +- will automatically be converted to \pm internally.

9.99(9)	`\num{9.99(9)}       \\`
9.99(9)	`\num{9.99 +- 0.09}  \\`
9.99(9)	`\num{9.99 \pm 0.09} \\`
123.0(45)	`\num{123 +- 4.5}    \\`
12.3(60)	`\num{12.3 +- 6}`

parse-numbers

The parse-numbers option turns the entire parsing system on and off. The option is made available for two reasons. First, if all of the numbers in a document are to be reproduced "as given", turning off the parser will represent a significant saving in processing required. Second, it allows the use of arbitrary TEX code in numbers. If the parser is turned off, the input will be printed in math mode (requiring \text to protect any text in the number).

```
\num[parse-numbers = false]{\sqrt{2}}        \\
\qty[parse-numbers = false]{\sqrt{3}}{\metre}
```

$\sqrt{2}$
$\sqrt{3}$ m

evaluate-expression
expression

With the standard settings, numerical input is parsed "as is" with no attempt to interpret it mathematically. By enabling the evaluate-expression option, the input can be processed by the standard LATEX3 FPU (see package xfp for more). The nature of the expression itself can be adjusted using the expression setting: as standard, the entire input is simply parsed with no change, but this setting may be used to add additional steps. The *input* in such an expression is represented by #1. Note that the FPU uses its own syntax for numbers, most notably in that a decimal marker must be ..

```
\sisetup{evaluate-expression}%
\qty{2 + 4 * 3}{\joule} \\
\qty[expression = 10 * (#1)]{2 + 4 * 3}{\joule}
```

14 J
140 J

3.4 Post-processing numbers

Before typesetting numbers, various post-processing steps can be carried out. These involve adding or removing information from the number in a systematic way; the options are summarised in Table 13.

exponent-mode
fixed-exponent

Numbers can be converted to scientific notation by the package. This is controlled by the exponent-mode option, which takes choices input, fixed, engineering and scientific. The fixed setting will use the exponent value by the fixed-exponent option. When engineering is set, the exponent is always a power of three.

Table 13: Number post-processing options.

Option name	Type	Default
`drop-exponent`	Switch	`false`
`drop-uncertainty`	Switch	`false`
`drop-zero-decimal`	Switch	`false`
`exponent-mode`	Switch	`input`
`fixed-exponent`	Integer	`0`
`minimum-integer-digits`	Integer	`0`
`minimum-decimal-digits`	Integer	`0`
`round-half`	Choice	`up`
`round-minimum`	Literal	`0`
`round-mode`	Choice	`none`
`round-pad`	Switch	`true`
`round-precision`	Integer	`2`

0.001
0.010 0
1 200
10^{-3}
1.00×10^{-2}
1.200×10^{3}
10^{-3}
10.0×10^{-3}
1.200×10^{3}
$0.000\,01 \times 10^{2}$
$0.000\,100 \times 10^{2}$
12.00×10^{2}

```
\num{0.001}   \\
\num{0.0100} \\
\num{1200}    \\
\sisetup{exponent-mode = scientific}%
\num{0.001}   \\
\num{0.0100} \\
\num{1200}    \\
\sisetup{exponent-mode = engineering}%
\num{0.001}   \\
\num{0.0100} \\
\num{1200}    \\
\sisetup{
  exponent-mode  = fixed,
  fixed-exponent = 2,
}%
\num{0.001}   \\
\num{0.0100} \\
\num{1200}
```

When used with a `fixed-exponent` of zero, this may be used to remove scientific notation from the input

```
\num{1.23e4} \\
\num[exponent-mode = fixed, fixed-exponent = 0]{1.23e4}
```

1.23×10^{4}
12 300

Exponent mode applies *after* rounding, such that the number of decimal places for rounding is those which appear in the output.

`drop-exponent`
`drop-uncertainty`

The use of an uncertainty can be suppressed entirely using the `drop-uncertainty` option: this applies *before* rounding is attempted. Similarly, exponents can be dropped using `drop-exponent` can be used to suppress the exponent part (*after* conversion to a fixed exponent).

0.01(2)
0.01
0.01×10^3
0.01

```
\num{0.01(2)} \\
\num[drop-uncertainty]{0.01(2)} \\
 \num{0.01e3} \\
\num[drop-exponent]{0.01e3}
```

`round-mode`
`round-precision`
`round-pad`

The package can round numerical input to a fixed number of significant figures or decimal places. This is controlled by the `round-mode` option, which takes the choices `none`, `figures`, `places` and `uncertainty`. When rounding is turned on, the number of digits used (either decimal places or significant figures in the mantissa) is set using the `round-precision` option. Rounding numbers with uncertainties may be carried out using the `uncertainty` setting to `round-mode`. In this case the precision is used first to round the uncertainty itself (to a number of figures), before rounding the main value to follow.

1.234 56
14.23
0.123 45(9)
1.235
14.230
0.123 45(9)
1.23
14.2
0.123 45(9) 0.123 45(9) 0.123 5(2) 0.123(2)

```
\num{1.23456} \\
\num{14.23} \\
\num{0.12345(9)} \\
\sisetup{
  round-mode      = places,
  round-precision = 3
}%
\num{1.23456} \\
\num{14.23} \\
\num{0.12345(9)} \\
\sisetup{
  round-mode      = figures,
  round-precision = 3
}%
\num{1.23456} \\
\num{14.23} \\
\num{0.12345(9)}
\sisetup{
  round-mode      = uncertainty,
  round-precision = 1
}%
\num{0.12345(9)}
\num{0.12345(23)}
\num{0.12345(234)}
```

Rounding my "extend" a short number to more digits (or figures): this is controlled by the switch `round-pad`, which is `true` as standard.

```
\sisetup{round-mode = figures, round-precision = 4}%
\num{12.3} \\
\num[round-pad = false]{12.3}
```

12.30
12.3

`round-half`

In cases where the rounded part of a number is exactly half, there are two common methods for "breaking the tie". The choice of method is determined by the option `round-half`, which recognises the choices `up` and `even`.

0.06
0.05
0.05
0.05

```
\sisetup{
  round-mode      = figures,
  round-precision = 1,
  round-half      = up
}%
\num{0.055} \\
\num{0.045} \\
\sisetup{round-half = even}%
\num{0.055} \\
\num{0.045}
```

`round-minimum` There are cases in which rounding will result in the number reaching zero. It may be desirable to show such results as below a threshold value. This can be achieved by setting `round-minimum` to the threshold value. There will be no effect when rounding to a number of significant figures as it is not possible to obtain the value zero in these cases.

0.01
0.00
0.01
0.00

```
\sisetup{round-mode = places}%
\num{0.0055} \\
\num{0.0045} \\
\sisetup{round-minimum = 0.01}% TODO
\num{0.0055} \\
\num{0.0045}
```

`drop-zero-decimal` It may be desirable to convert decimals to integers when the decimal part is zero. This is set up using the `drop-zero-decimal` option, which applies after rounding but before setting minimum numbers of digits.

2.0
2.1
2
2.1

```
\num{2.0} \\
\num{2.1} \\
\sisetup{drop-zero-decimal}%
\num{2.0} \\
\num{2.1}
```

`minimum-decimal-digits`
`minimum-integer-digits`
The `minimum-decimal-digits` and `minimum-integer-digits` option may be used to pad numbers to a given size. This applies independent of any rounding.

```
\num{123} \\
\num[minimum-integer-digits = 2]{123} \\
\num[minimum-integer-digits = 4]{123} \\
\num{0.123} \\
\num[minimum-decimal-digits = 2]{0.123} \\
\num[minimum-decimal-digits = 4]{0.123} \\
```

123
123
0 123
0.123
0.123
0.123 0

Table 14: Output options for numbers.

Option name	Type	Default
`bracket-negative`	Switch	`false`
`close-bracket`	Literal	`)`
`exponent-base`	Literal	`10`
`exponent-product`	Math	`\times`
`group-digits`	Choice	`all`
`group-minimum-digits`	integer	`5`
`group-separator`	Literal	`\,`
`negative-color`	Literal	⟨*none*⟩
`open-bracket`	Literal	`(`
`output-close-uncertainty`	Literal	`)`
`output-decimal-marker`	Literal	`.`
`output-open-uncertainty`	Literal	`(`
`separate-uncertainty`	Switch	`false`
`tight-spacing`	Switch	`false`
`uncertainty-separator`	Literal	⟨*none*⟩

3.5 Printing numbers

Actually printing numbers is controlled by a number of settings, which apply ideas such as differing decimal markers, digit grouping and so on. All of these options are concerned with the appearance of output, rather than the data it conveys. The options are summarised in Table 14.

`group-digits`
`group-four-digits`
`group-separator`

Grouping digits into blocks of three is a common method to increase the ease of reading of numbers. The `group-digits` choice controls whether this behaviour applies, and takes the values `all`, `none`, `decimal` and `integer`. Grouping can be activated separately for the integer and decimal parts of a number using the appropriately-named values.

```
\num{12345.67890} \\
\num[group-digits = none]{12345.67890}   \\
\num[group-digits = decimal]{12345.67890} \\
\num[group-digits = integer]{12345.67890}
```

12 345.678 90
12345.67890
12345.678 90
12 345.67890

The separator used between groups of digits is stored by the `group-separator` option. This takes literal input and may be used in math mode: for a text-mode full space use `\␣`.

12 345
12, 345
12 345

```
\num{12345} \\
\num[group-separator = {,}]{12345} \\
\num[group-separator = \ ]{12345}
```

`group-minimum-digits`

Grouping is not always applied to smaller numbers, and the option `group-minimum-digits` is available to specify how many digits must be present before grouping is applied. The number of digits is considered separately for the integer and decimal parts of the number: grouping does not "cross the boundary".

```
\num{1234} \\
\num{12345} \\
\num[group-minimum-digits = 5]{1234} \\
\num[group-minimum-digits = 5]{12345} \\
\num{1234.5678} \\
\num{12345.67890} \\
\num[group-minimum-digits = 5]{1234.5678} \\
\num[group-minimum-digits = 5]{12345.67890}
```

1 234
12 345
1234
12 345
1 234.567 8
12 345.678 90
1234.5678
12 345.678 90

output-decimal-marker

The decimal marker used in output is set using the `output-decimal-marker` option; this can differ from the input marker.

```
\num{1.23} \\
\num[output-decimal-marker = {,}]{1.23} \\
```

1.23
1,23

exponent-base
exponent-product

When exponents are present in the input, the `exponent-base` and `exponent-product` options set the obvious parts of the output.

```
\num[exponent-product = \times]{1e2} \\
\num[exponent-product = \cdot]{1e2} \\
\num[exponent-base = 2]{1e2}
```

10^2
10^2
2^2

separate-uncertainty
uncertainty-separator
output-open-uncertainty
output-close-uncertainty

When input is given including an uncertainty in a number, it can be printed either with the uncertainty in brackets or as a separate number. This behaviour is controlled by the `separate-uncertainty` choice. If the uncertainty is given in brackets, a space may be added between the main number and the uncertainty: this is stored using the `uncertainty-separator` option. The opening and closing brackets used are stored in `output-open-uncertainty` and `output-close-uncertainty`, respectively.

```
\num{1.234(5)} \\
\num[separate-uncertainty = true]{1.234(5)} \\
\sisetup{
  output-open-uncertainty  = [,
  output-close-uncertainty = ],
  uncertainty-separator    = \,
}%
\num{1.234(5)}
```

1.234(5)
1.234 ± 0.005
1.234 [5]

Table 15: Angle options.

Option name	Type	Default
`angle-mode`	Choice	`input`
`angle-symbol-over-decimal`	Switch	`false`
`arc-separator`	Literal	⟨*empty*⟩
`fill-arc-degrees`	Switch	`false`
`fill-arc-minutes`	Switch	`false`
`fill-arc-seconds`	Switch	`false`
`number-angle-product`	Literal	⟨*empty*⟩

Notice that siunitx correctly interprets uncertainties which cross the decimal marker position whether these are separated out or not.

8.2(13)
8.2 ± 1.3

```
\num{8.2(13)} \\
\num[separate-uncertainty]{8.2(13)}
```

`negative-color` The package can detect negative mantissa values and alter print color accordingly. This is disabled by setting the option to an empty value.

−15 673
−15 673

```
\num{-15673} \\
\num[negative-color = red]{-15673}
```

`bracket-negative` A common means to display negative numbers in financial situations is to place them in brackets. This can be carried out automatically using the `bracket-negative-numbers` option.

−15 673
(15 673)
−10 m
(10) m

```
\num{-15673} \\
\num[bracket-negative]{-15673} \\
\qty{-10}{\metre} \\
\qty[bracket-negative]{-10}{\metre}
```

`tight-spacing` Under some circumstances is may be desirable to "squeeze" the output spacing. This is turned on using the `tight-spacing` switch, which compresses spacing where possible.

2×10^3
$2{\times}10^3$

```
\num{2e3} \\
\num[tight-spacing = true]{2e3}
```

3.6 Angles

Angle processing provided by the `\ang` function has a set of options which apply in addition to the general ones set up for number processing.

`angle-mode` The format in which angles are printed can be set using the `angle-mode` option. With the standard setting (`input`), the angle is printed as-given. By setting the option to `arc` or `decimal`, the output format can be set to an arc (degrees/minutes/seconds) or decimal value. Conversion uses the LaTeX3 floating-point unit, so is limited to 16 decimal places.

2.67°
2°3′4″
2°40′12″
2°3′4″
2.67°
2.051 111 111 111 111°

```
\ang{2.67} \\
\ang{2;3;4} \\
\ang[angle-mode = arc]{2.67} \\
\ang[angle-mode = arc]{2;3;4} \\
\ang[angle-mode = decimal]{2.67} \\
\ang[angle-mode = decimal]{2;3;4} \\
```

`number-angle-product`

The separator between the number and angle symbol (degrees, minutes or seconds) can be set using the `number-angle-product` option, independent of the related `number-unit-product` option used by the `\qty` command.

2.67°
2.67 °

```
\ang{2.67} \\
\ang[number-angle-product = \,]{2.67}
```

`arc-separator`

When angles are printed in arc format, the separation of the different parts is set up using the `arc-separator` option.

6°7′6.5″
6° 7′ 6.5″

```
\ang{6;7;6.5} \\
\ang[arc-separator = \,]{6;7;6.5}
```

`fill-arc-degrees`
`fill-arc-minutes`
`fill-arc-seconds`

Zero-filling for the degree, minute or second parts of an arc is controlled using the `fill-arc-degrees`, `fill-arc-minutes` and `fill-arc-seconds` options. All are off as standard.

−1°
−2′
−3″
−1°
−0°2′
−0°3″
−1°0′
−2′
−0′3″
−1°0″
−2′0″
−3″

```
\ang{-1;;} \\
\ang{;-2;} \\
\ang{;;-3} \\
{
  \sisetup{fill-arc-degrees}
  \ang{-1;;} \\
  \ang{;-2;} \\
  \ang{;;-3} \\
}
{
  \sisetup{fill-arc-minutes}
  \ang{-1;;} \\
  \ang{;-2;} \\
  \ang{;;-3} \\
}
{
  \sisetup{fill-arc-seconds}
  \ang{-1;;} \\
  \ang{;-2;} \\
  \ang{;;-3}
}
```

`angle-symbol-over-decimal`

In some subject areas, most notably astronomy, the angle symbols are given over the decimal marker, rather than at the end of the number. This behaviour is available using the `angle-symbol-over-decimal` option.

Table 16: Unit output options.

Option name	Type	Default
bracket-unit-denominator	Switch	true
font-command	Literal	\mathrm
forbid-literal-units	Switch	false
inter-unit-product	Literal	\,
parse-units	Switch	true
per-mode	Choice	power
per-symbol	Literal	/
qualifier-mode	Choice	subscript
qualifier-phrase	Literal	⟨*empty*⟩
sticky-per	Switch	false

3.7 Using units

Part of the power of siunitx is the ability to alter the output format for units without changing the input. The behaviour of units is therefore controlled by a number of options which alter either the processing of units or the output directly (Table 16).

inter-unit-product

The separator between each unit is stored using the inter-unit-product option. The standard setting is a thin space: another common choice is a centred dot. To get the correct spacing it is necessary to use \ensuremath{{}\cdot{}} in the latter case.

```
\unit{\farad\squared\lumen\candela} \\
\unit[inter-unit-product = \ensuremath{{}\cdot{}}]
  {\farad\squared\lumen\candela}
```

$\mathrm{F^2\,lm\,cd}$
$\mathrm{F^2\cdot lm\cdot cd}$

per-mode
per-symbol
bracket-unit-denominator

The handling of \per is altered using the per-mode choice option. The standard setting is power, meaning that \per generates reciprocal powers for units. Setting the option to fraction uses the \frac function to typeset the positive and negative powers of a unit separately.

```
\unit{\joule\per\mole\per\kelvin} \\
\unit{\metre\per\second\squared} \\
\unit[per-mode = fraction]{\joule\per\mole\per\kelvin} \\
\unit[per-mode = fraction]{\metre\per\second\squared}
```

$\mathrm{J\,mol^{-1}\,K^{-1}}$
$\mathrm{m\,s^{-2}}$
$\frac{\mathrm{J}}{\mathrm{mol\,K}}$
$\frac{\mathrm{m}}{\mathrm{s^2}}$

The closely-related power-positive-first setting acts in the same way but places all of the positive powers before any negative ones.

$\mathrm{A\,mol^{-1}\,s}$
$\mathrm{A\,s\,mol^{-1}}$

```
\unit{\ampere\per\mole\second} \\
\unit[per-mode = power-positive-first]
  {\ampere\per\mole\second}
```

It is possible to use a symbol (usually `/`) to separate the two parts of a unit by setting `per-mode` to `symbol`; the symbol used is stored using the setting `per-symbol`. This method for displaying units can be ambiguous, and so brackets are added unless `bracket-unit-denominator` is set to `false`. Notice that `bracket-unit-denominator` only applies when `per-mode` is set to `symbol` or `symbol-or-fraction`.

```
\sisetup{per-mode = symbol}%
\unit{\joule\per\mole\per\kelvin} \\
\unit{\metre\per\second\squared} \\
\unit[per-symbol = \ \text{div}\ ]{\joule\per\mole\per\kelvin} \\
\unit[bracket-unit-denominator = false]{\joule\per\mole\per\kelvin}
```

J/(mol K)
m/s^2
J div (mol K)
J/mol K

The often-requested (but mathematically invalid) `repeated-symbol` option is also available to repeat the symbol for each `\per`.

```
\unit[per-mode = repeated-symbol]{\joule\per\mole\per\kelvin}
```

J/mol/K

Finally, it is possible for the behaviour of the `\per` function to depend on the prevailing math style. Setting `per-mode` to `symbol-or-fraction` will use the `symbol` setting for in line math, and the `fraction` setting when used in display math.

```
\sisetup{per-mode = symbol-or-fraction}%
$ \unit{\joule\per\mole\per\kelvin} $
\[ \unit{\joule\per\mole\per\kelvin} \]
   \unit{\joule\per\mole\per\kelvin} \\
$
  \displaystyle
  \unit{\joule\per\mole\per\kelvin}
$
\[
  \textstyle
  \unit{\joule\per\mole\per\kelvin}
\]
```

J/(mol K)

$$\frac{\mathrm{J}}{\mathrm{mol\,K}}$$

J/(mol K)
$\dfrac{\mathrm{J}}{\mathrm{mol\,K}}$

$$\mathrm{J/(mol\,K)}$$

`sticky-per` By default, `\per` applies only to the next unit given.[2] By setting the `sticky-per` flag, this behaviour is changed so that `\per` applies to all subsequent units.

[2] This is the standard method of reading units in English: for example, $\mathrm{J\,mol^{-1}\,K^{-1}}$ is pronounced "joules per mole per kelvin".

```
\unit{\pascal\per\gray\henry} \\
\unit[sticky-per]{\pascal\per\gray\henry}
```
Pa Gy^{-1} H
Pa Gy^{-1} H^{-1}

qualifier-mode
qualifier-phrase

Unit qualifiers can be printed in three different formats, set by the `qualifier-mode` option. The standard setting is `subscript`, while the options `bracket`, `combine` and `phrase` are also possible. With the last settings, powers can lead to ambiguity and are automatically detected and brackets added as appropriate.

```
\unit{\kilogram\of{pol}\squared\per\mole\of{cat}\per\hour} \\
\unit[qualifier-mode = bracket]
  {\kilogram\of{pol}\squared\per\mole\of{cat}\per\hour} \\
\unit[qualifier-mode = combine]
  {\deci\bel\of{i}}
```
$\mathrm{kg}_{\mathrm{pol}}^{2}\,\mathrm{mol}_{\mathrm{cat}}^{-1}\,\mathrm{h}^{-1}$
$\mathrm{kg(pol)}^{2}\,\mathrm{mol(cat)}^{-1}\,\mathrm{h}^{-1}$
dBi

The `phrase` option is used with `qualifier-phrase`, which allows for example a space or other linking text to be inserted.

```
\sisetup{qualifier-mode = phrase, qualifier-phrase = \ }%
\unit{\kilogram\of{pol}\squared\per\mole\of{cat}\per\hour} \\
\sisetup{qualifier-phrase = \ \mbox{of}\ }%
\unit{\kilogram\of{pol}\squared\per\mole\of{cat}\per\hour}
```
kg pol^{2} mol cat^{-1} h^{-1}
kg of pol^{2} mol of cat^{-1} h^{-1}

parse-units

Normally, siunitx is used with the unit parse enabled, and only prints units directly if there is literal input. However, if the input is known to be essentially consistent and high performance is desired, then the parser can be turned off using the `parse-units` switch.

300 MHz	`\qty{300}{\MHz} \\`
300 MHz	`\qty[parse-units = false]{300}{\MHz}`

forbid-literal-units

Some users may prefer to completely disable the use of literal input in units, for example to enforce consistency. This can be accomplished by setting the `forbid-literal-units` switch. With this option enabled, only macro-based units can be used in a document. This only applies when `parse-units` is `true`.

font-command

The command used to set unit themselves may be adjusted using the `font-command` option. This is typically set to `\mathrm`.

lm	`\unit{\lumen} \\`
lm	`\unit[font-command = \mathit]{\lumen}`

Table 17: Options for tabular material.

Option name	Type	Default
`table-align-comparator`	Switch	`true`
`table-align-exponent`	Switch	`true`
`table-align-text-after`	Switch	`true`
`table-align-text-before`	Switch	`true`
`table-align-uncertainty`	Switch	`true`
`table-alignment`	Meta	`center`
`table-alignment-mode`	Choice	`marker`
`table-auto-round`	Switch	`false`
`table-column-width`	Length	`0pt`
`table-fixed-width`	Switch	`false`
`table-format`	Special	`2.2`
`table-number-alignment`	Choice	`center`
`table-text-alignment`	Choice	`center`

3.8 Tabular material

Processing of material in tables obeys the same settings as described for the functions already described. However, there are some settings which apply only to the layout of tabular material (Table 17).

`table-mode` The method used by siunitx to align numbers is selected using the `table-mode` option, which may be one of `marker`, `format` or ⟨*none*⟩. With the standard setting, ⟨*marker*⟩, the package centers the decimal marker in a tabular column, potentially leaving white space at the shorter end of a number. The ⟨*format*⟩ mode uses information from the ⟨*table-format*⟩ key to construct a model: this is then used to define the space available to a number. For asymmetrical numbers, this method is strongly preferable. Finally, `none` disables alignment entirely: numbers are simply parsed.

`table-number-alignment` When `table-mode` is set to ⟨*format*⟩ or ⟨*none*⟩, the placement of the number "block" within the cell as a whole is set by the ⟨*table-number-alignment*⟩ option, which may be one of ⟨*left*⟩, ⟨*center*⟩ or ⟨*right*⟩. (When `table-mode` is set to ⟨*marker*⟩, the decimal marker is always centered in the cell.) The different alignment choices are illustrated in Table 18, which uses somewhat exaggerated column headings to show the relative position of the cell contents.

```
\begin{table}
  \caption{Aligning the \texttt{S} column.}
  \label{tab:S:align}
  \centering
  \sisetup{table-format = 2.4, table-alignment-mode = format}
  \begin{tabular}{@{}
    S[table-alignment-mode = marker]
    S[table-number-alignment = center]
    S[table-number-alignment = left]
    S[table-number-alignment = right]
  @{}}
  \toprule
  {Some Values} & {Some Values} & {Some Values} & {Some Values} \\
  \midrule
```

Table 18: Aligning the `S` column.

Some Values	Some Values	Some Values	Some Values
2.345 6	2.345 6	2.345 6	2.345 6
34.234 5	34.234 5	34.234 5	34.234 5
56.783 5	56.783 5	56.783 5	56.783 5
90.473	90.473	90.473	90.473

Table 19: Parsing without aligning in an `S` column.

Decimal-centred	Simple centring
12.345	12.345
6.78	6.78
−88.8(9)	−88.8(9)
4.5×10^3	4.5×10^3

```
     2.3456  &  2.3456  &  2.3456 &  2.3456 \\
    34.2345  & 34.2345  & 34.2345 & 34.2345 \\
    56.7835  & 56.7835  & 56.7835 & 56.7835 \\
    90.473   & 90.473   & 90.473  & 90.473  \\
  \bottomrule
  \end{tabular}
\end{table}
```

When the alignment mode is set to `none`, number are simply collected and parsed without any further processing, as illustrated in Table 19.

```
\begin{table}
  \caption{Parsing without aligning in an \texttt{S} column.}
  \label{tab:S:parse}
  \begin{tabular}
    {@{}
      S
      S[table-alignment-mode = none]
    @{}}
    \toprule
      {Decimal-centred} &
      {Simple centring} \\
    \midrule
      12.345   & 12.345   \\
      6,78     & 6,78     \\
      -88.8(9) & -88.8(9) \\
      4.5e3    & 4.5e3    \\
    \bottomrule
  \end{tabular}
\end{table}
```

`table-format` When the `table-alignment-mode` is set to `format`, siunitx uses the information set in ⟨*table-format*⟩ to construct a "model" which defines the space to reserve for a number. The `table-format` key is interpreted in much the same way as a table cell. The numerical part should consist of a number showing how many figures to reserve in each part of the

Table 20: Reserving space in `S` columns.

Values	Values	Values	Values	Values	Values
2.3	2.3	2.3(5)	2.3 ± 0.5	2.3	2.3×10^8
34.23	34.23	34.23(4)	34.23 ± 0.04	34.23	34.23
56.78	56.78	56.78(3)	56.78 ± 0.03	-56.78	56.78×10^3
3.76	3.76	3.76(2)	3.76 ± 0.02	± 3.76	10^6

input, plus any comparators, signs, *etc.* A variety of examples are given in Table 20.

```
\begin{table}
  \caption{Reserving space in \texttt{S} columns.}
  \label{tab:S:format}
  \sisetup{
    table-alignment-mode   = format,
    table-number-alignment = center,
  }
  \begin{tabular}{@{}
    S[table-format = 2.2]
    S[table-format = 2.2, table-number-alignment = right]
    S[table-format = 2.2(1)]
    S[table-format = 2.2(1), separate-uncertainty]
    S[table-format = +2.2]
    S[table-format = 2.2e1]
   @{}}
  \toprule
      {Values}
    & {Values}
    & {Values}
    & {Values}
    & {Values}
    & {Values} \\
  \midrule
     2.3  &  2.3  &  2.3(5)  &  2.3(5)  &   2.3  &  2.3e8  \\
    34.23 & 34.23 & 34.23(4) & 34.23(4) &  34.23 & 34.23   \\
    56.78 & 56.78 & 56.78(3) & 56.78(3) & -56.78 & 56.78e3 \\
     3,76 &  3,76 &  3,76(2) &  3.76(2) & +-3.76 &      e6 \\
  \bottomrule
  \end{tabular}
\end{table}
```

It is important to note that any parts of a number *not* specified in the table format argument are set to be absent (the number of figures is set to zero). Setting the `table-format` option also resets `table-alignment-mode` to `format`.

Space for material before and after the `S` column can be reserved by giving model text as part of the `table-format` key. This is then used to provide the necessary gap while maintaining alignment (Table 21).

```
\begin{table}
  \caption{Text before and after numbers.}
  \label{tab:S:ends}
  \sisetup{table-format = {now }2.4{\textsuperscript{\emph{a}}}}
  \begin{tabular}{@{}S@{}}
```

Table 21: Text before and after numbers.

Values
2.345 6
34.234 5[a]
56.783 5
now 90.473

Table 22: The `table-align-exponent` option

Header	Header
$1.2 \quad \times 10^{3}$	1.2×10^{3}
1.234×10^{56}	1.234×10^{56}

```
  \toprule
  {Values} \\
  \midrule
          2.3456  \\
         34.2345  \textsuperscript{\emph{a}}\\
         56.7835  \\
    now~ 90.473   \\
  \bottomrule
  \end{tabular}
\end{table}
```

`table-align-exponent`
`table-align-uncertainty`

When printing exponents in tables, there is a choice of aligning the exponent parts or having these close up to the mantissa. This is controlled by the `table-align-exponent` option (Table 22). Similarly, uncertainty parts which are printed separately from the mantissa can be aligned or closed up. This is set by the `table-align-uncertainty` option (Table 23). Finally, the same approach is available for the comparator with the `table-align-comparator` option (Table 24).

```
\begin{table}
  \caption{The \opt{table-align-exponent} option}
  \label{tab:align:exp}
  \sisetup{table-format = 1.3e2}
  \begin{tabular}{@{}SS[table-align-exponent = false]@{}}
    \toprule
      {Header} & {Header} \\
    \midrule
      1.2e3    & 1.2e3    \\
      1.234e56 & 1.234e56 \\
    \bottomrule
  \end{tabular}
\end{table}

\begin{table}
  \caption{The \opt{table-align-uncertainty} option}
  \label{tab:align:uncert}
  \sisetup{
    separate-uncertainty,
```

Table 23: The `table-align-uncertainty` option

Header	Header
1.2 ± 0.1	1.2 ± 0.3
1.234 ± 0.005	1.234 ± 0.005

Table 24: The `table-align-comparator` option

Header	Header
> 1.2	>1.2
<12.34	<12.34

```
    table-format = 1.3(1),
  }
  \begin{tabular}{@{}SS[table-align-uncertainty = false]@{}}
    \toprule
      {Header} & {Header} \\
    \midrule
      1.2(1)   & 1.2(3)    \\
      1.234(5) & 1.234(5) \\
    \bottomrule
  \end{tabular}
\end{table}

\begin{table}
  \caption{The \opt{table-align-comparator} option}
  \label{tab:align:comp}
  \sisetup{table-format = >2.2}
  \begin{tabular}{@{}SS[table-align-comparator = false]@{}}
    \toprule
      {Header} & {Header} \\
    \midrule
      >  1.2  & >  1.2  \\
      < 12.34 & < 12.34 \\
    \bottomrule
  \end{tabular}
\end{table}
```

`table-align-text-before`
`table-align-text-after`

Note markers are often given in tables after the numerical content. It may be desirable for these to close up to the numbers. Whether this takes place is controlled by the `table-align-text-before` and `...-after` option (Table 25).

```
\begin{table}
  \caption{Closing notes up to text.}
  \label{tab:S:notes}
  \newrobustcmd\NoteMark[1]{%
    \textsuperscript{\emph{#1}}%
  }
  \sisetup{table-format = {\NoteMark{a}}2.4}
  \begin{tabular}{@{}
    S
```

Table 25: Closing notes up to text.

Values	Values
2.345 6	2.345 6
^{a} 4.234	^{a}4.234
^{b} 0.78	^{b}0.78
^{d}88	^{d}88

Values		Values
2.345 6		2.345 6
34.234	^{a}	34.234^{a}
56.78	^{b}	56.78^{b}
90.4	^{c}	90.4^{c}
88	^{d}	88^{d}

```
    S[table-align-text-before = false]
  @{}}
  \toprule
    {Values}             & {Values} \\
  \midrule
                2.3456 &               2.3456 \\
    \NoteMark{a}  4.234  & \NoteMark{a}  4.234 \\
    \NoteMark{b}    .78  & \NoteMark{b}    .78  \\
    \NoteMark{d} 88      & \NoteMark{d} 88      \\
  \bottomrule
  \end{tabular}
  \hfil
  \sisetup{table-format = 2.4\NoteMark{a}}
  \begin{tabular}{
    S
    S[table-align-text-after = false]
  }
  \toprule
    {Values}             & {Values} \\
  \midrule
     2.3456              & 2.3456 \\
    34.234  \NoteMark{a} & 34.234 \NoteMark{a} \\
    56.78   \NoteMark{b} & 56.78  \NoteMark{b} \\
    90.4    \NoteMark{c} & 90.4   \NoteMark{c} \\
    88      \NoteMark{d} & 88     \NoteMark{d} \\
  \bottomrule
  \end{tabular}
\end{table}
```

`table-auto-round`

The contents of table cells can automatically be rounded or zero-filled to the number of decimal digits given for the decimal part of the `table-format` option. This mode is activated using the `table-auto-round` switch, as illustrated in Table 26.

```
\begin{table}
  \centering
  \caption{The \opt{table-auto-round} option.}
  \label{tab:S:auto}
  \sisetup{table-format = 1.3}
  \begin{tabular}{@{}SS[table-auto-round]@{}}
    \toprule
      {Header} & {Header} \\
    \midrule
      1.2    & 1.2    \\
      1.2345 & 1.2345 \\
```

Table 26: The `table-auto-round` option.

Header	Header
1.2	1.200
1.234 5	1.235

```
    \bottomrule
  \end{tabular}
\end{table}
```

parse-numbers

When the `parse-numbers` option is set to `false`, then the alignment code for tables takes a different approach. The output is always set in math mode, and alignment takes place at the first decimal marker. This is achieved by making it active in math mode. When reserving space for content only the integer and decimal values for the mantissa are considered (Table 27).

```
\begin{table}
  \caption{Aligning without parsing.}
  \label{tab:S:nonparsed}
  \sisetup{
    parse-numbers = false,
    table-format  = 3.3
  }
  \centering
  \begin{tabular}{@{}
    S
    S[table-number-alignment = center]
    S[table-number-alignment = right]
    S[table-number-alignment = left]
    @{}}
  \toprule
      {Some values}
    & {Some values}
    & {Some values}
    & {Some values} \\
  \midrule
     2.35  &    2.35  &    2.35  &   2.35   \\
    34.234 &   34.234 &   34.234 &  34.234  \\
    56.783 &   56.783 &   56.783 &  56.783  \\
     3,762 &    3,762 &    3,762 &   3.762  \\
  \sqrt{2} & \sqrt{2} & \sqrt{2} & \sqrt{2} \\
  \bottomrule
  \end{tabular}
\end{table}
```

drop-exponent

In cases where data cover a range of values, printing using a fixed exponent in a table may make presentation clearer. In these cases, dropping the exponent value from the table is useful. The general numerical options `drop-exponent` combined with `exponent-mode = fixed` can be used to achieve this (Table 28).

```
\begin{table}
  \caption{The \opt{table-omit-exponent} option}
  \label{tab:exp:omit}
```

Table 27: Aligning without parsing.

Some values	Some values	Some values	Some values
2.35	2.35	2.35	2.35
34.234	34.234	34.234	34.234
56.783	56.783	56.783	56.783
3.762	3.762	3.762	3.762
$\sqrt{2}$	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{2}$

Table 28: The `table-omit-exponent` option

Header	Header / 10^3
1.2×10^3	1.2
$3 \quad \times 10^2$	0.3
1.0×10^4	10

```
  \begin{tabular}{@{}
    S[table-format = 1.1e1]
    S[
      drop-exponent  = true,
      exponent-mode  = fixed,
      fixed-exponent = 3,
      table-format   = 2.1,
    ]
  @{}}
    \toprule
      {Header} & \multicolumn{1}{c@{}}{Header / \num{e3}} \\
    \midrule
      1.2e3 & 1.2e3 \\
       3e2  &   3e2 \\
     1.0e4  & 1.0e4 \\
    \bottomrule
  \end{tabular}
\end{table}
```

`table-column-width`
`table-fixed-width`

Usually, the width of the numerical column is allowed to vary depending on the content. However, there are cases where a strictly fixed width is desirable. For these cases, the `table-fixed-width` and `table-column-width` options are available. The `table-fixed-width` option activates fixed-width columns, whilst `table-column-width` defines the target width (Table 29).

```
\begin{table}
  \caption{Fixed-width columns.}
  \label{tab:width:fixed}
  \begin{tabular}
    {@{}
      S
      S[table-column-width = 2cm]
    @{}}
    \toprule
      {Flexible} &
```

Table 29: Fixed-width columns.

Flexible	Fixed
1.23	1.23
45.6	45.6

Table 30: Right-aligning under a heading.

Long header
12.33
2
1 234

```
      {Fixed}    \\
    \midrule
      1.23 & 1.23 \\
      45.6 & 45.6 \\
    \bottomrule
  \end{tabular}
\end{table}
```

The `table-column-width` option can also be used to achieve special effects. One example is centring a column of numbers under a wide heading, with the numbers themselves right-aligned (Table 30).

```
\begin{table}
  \centering
  \caption{Right-aligning under a heading.}
  \label{tab:width:special}
  \settowidth{\mylength}{Long header}
  \sisetup{
    table-alignment-mode   = none      ,
    table-column-width     = \mylength ,
    table-number-alignment = right
  }
  \begin{tabular}{@{}S@{}}
  \toprule
  {Long header} \\
  \midrule
      12.33 \\
       2    \\
    1234    \\
   \bottomrule
  \end{tabular}
\end{table}
```

`table-text-alignment`
`table-alignment`

Cell contents which are not part of a number can be protected using braces, as illustrated. Cells which contain no numerical data at all are aligned using the setting specified by the `table-text-alignment` option, which recognises the values `center`, `left` and `right` (Table 31).

```
\begin{table}
```

Table 31: Aligning text in S columns.

Values	Values	Values
992.435	992.435	992.435
7 734.234 4	7 734.234 4	7 734.234 4
56.783 4	56.783 4	56.783 4
3.746 2	3.746 2	3.746 2

```
  \caption{Aligning text in \texttt{S} columns.}
  \label{tab:S:text}
  \sisetup{table-format = 4.4}
  \centering
  \begin{tabular}{@{}
    S
    S[table-text-alignment = left]
    S[table-text-alignment = right]
  @{}}
  \toprule
      {Values}
    & {Values}
    & {Values} \\
  \midrule
    992.435  &  992.435  &  992.435  \\
   7734.2344 & 7734.2344 & 7734.2344 \\
     56.7834 &   56.7834 &   56.7834 \\
      3,7462 &    3,7462 &    3,7462 \\
  \bottomrule
  \end{tabular}
\end{table}
```

table-alignment The table alignment options `table-number-alignment` and `table-text-alignment` can be set to the same value using the `table-alignment` option. This will set all three alignment options to the same value (one of `center`, `right` or `left`).

4 Installation

For most users, there will be no need to explicitly install siunitx: it is available from the package management system in current TeX Live and MiKTeX systems.

For manual installation, the package is available from CTAN. As well as the raw source files, CTAN hold the package as a pre-extracted zip file, `siunitx.tds.zip`. The later is most convenient for most users: simply unzip this in your local `texmf` directory.

The package requires LaTeX3 support as provided in the l3kernel and l3packages bundles. Both of these are included in TeX Live and MiKTeX, or are again available in ready-to-install form from CTAN.

5 Thanks

Many users have provided feedback, bug reports and ideas for new features for siunitx: thanks to all of them. Particular thanks to Stefan Pinnow, who has taken the lead role as beta tester for siunitx, finding incorrect output, bad documentation and the odd spelling mistake in the documentation. Thanks also to Enrico Gregorio for encouraging me to complete a fully expl3-compliant version of the package. Thanks also to Danie Els and Marcel Heldoorn for the SIstyle and SIunits packages, respectively, which provided the starting point for the development of siunitx.

6 Making suggestions and reporting bugs

Feedback on siunitx is always welcome, either to make suggestions or to report problems. When sending feedback, it is always useful if a small example file is included, showing the bug being reported or illustrating the desired output. It is helpful if a "reference rendering" is included, showing what the output should look like. A typical example file might read

```
\listfiles
% Use the article class unless the problem is class-dependent
\documentclass{article}
\usepackage{siunitx}
% Other packages loaded as required
\begin{document}
Reference output: $1.23\,\mathrm{m}$

siunitx output: \qty{1.23}{\metre}
\end{document}
```

As illustrated, it is usually best to use the article class and to only load packages which are needed to show the issue. It is also useful to include a copy of the log file generate by LaTeX when reporting a bug (as the versions of packages can be important to solving the issue).

Feedback can be sent in a range of ways. The development code and issue tracker are hosted on GitHub. Issues opened there are visible to other users and makes sure that they cannot be forgotten.

References

[1] *The International System of Units (SI)*, https://www.bipm.org/en/measurement-units/.

[2] *SI base units*, https://www.bipm.org/en/publications/si-brochure/section2-1.html.

[3] *Units with special names and symbols; units that incorporate special names and symbols*, https://www.bipm.org/en/publications/si-brochure/section2-2-2.html.

[4] *SI Prefixes*, https://www.bipm.org/en/publications/si-brochure/chapter3.html.

[5] *Formatting the value of a quantity*, https://www.bipm.org/en/publications/si-brochure/section5-3-3.html.

[6] *Stating values of dimensionless quantities, or quantities of dimension one*, `https://www.bipm.org/en/publications/si-brochure/section5-3-7.html`.

[7] *Non-SI units accepted for use with the International System of Units*, `https://www.bipm.org/en/publications/si-brochure/table6.html`.

[8] *Non-SI units whose values in SI units must be obtained experimentally*, `https://www.bipm.org/en/publications/si-brochure/table7.html`.

[9] *Other non-SI units*, `https://www.bipm.org/en/publications/si-brochure/table8.html`.

[10] *Non-SI units associated with the CGS and the CGS-Gaussian system of units*, `https://www.bipm.org/en/publications/si-brochure/table9.html`.

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

P

Q

R

S

T