An Acronym Environment for LaTeX 2ε*

Tobias Oetiker

2020/04/17

1 Introduction

When writing a paper on cellular mobile radio I started to use a lot of acronyms. This can be very disturbing for the reader, as he might not know all the used acronyms. To help the reader I kept a list of all the acronyms at the end of my paper.

This package makes sure, that all acronyms used in the text are spelled out in full at least once.

2 The user interface

The package provides several commands and one environment for dealing with acronyms. Their appearance can be controlled by two package options and three macros.

2.1 Acronyms in the Text

`\ac` To enter an acronym inside the text, use the

`\ac[`⟨*linebreak penalty*⟩`]{`⟨*acronym*⟩`}`

command. The first time you use an acronym, the full name of the acronym along with the acronym in brackets will be printed. If you specify the `footnote` option while loading the package, the full name of the acronym is printed as a footnote. The next time you access the acronym only the acronym will be printed.

When an acronym is being used, for the first time (with the `footnote` option not specified), next to the end of the line, a line break between the full name of the acronym and the acronym in brackets can be encountered. The optional variable represents the penalty level of breaking the line at that place, taking integer values between 0 and 4. A higher number corresponds to a higher penalty.

`\Ac` Works in the same way as `\ac`, but starts the long form with an upper case

*This file has version number v1.47, last revised 2020/04/17.

letter. Use case: when the acronym is used for the first time, at the beginning of a sentence.

`\acresetall` The 'memory' of the macro `\ac` can be flushed by calling the macro `\acresetall`. Afterwards, `\ac` will print the full name of any acronym and the acronym in brackets the next time it is used.

`\acf` If later in the text again the Full Name of the acronym should be printed, use the command

`\acf[`⟨*linebreak penalty*⟩`]{`⟨*acronym*⟩`}`

to access the acronym. It stands for "full acronym" and it always prints the full name and the acronym in brackets.

When an full acronym is being used next to the end of the line, a line break between the full name of the acronym and the acronym in brackets can be encountered. The optional variable represents the penalty level of breaking the line at that place, taking integer values between 0 and 4. A higher number corresponds to a higher penalty.

`\Acf` Works in the same way as `\acf`, but starts the long form with an upper case letter.

`\acs` To get the short version of the acronym, use the command

`\acs{`⟨*acronym*⟩`}`

`\acl` Gives you the expanded acronym without even mentioning the acronym.

`\acl{`⟨*acronym*⟩`}`

`\Acl` Works in the same way as `\acl`, but starts with an upper case letter.

`\acp` Works in the same way as `\ac`, but makes the short and/or long forms into plurals.

`\Acp` Works in the same way as `\acp`, but starts the long form with an upper case letter.

`\acfp` Works in the same way as `\acf`, but makes the short and long forms into plurals.

`\Acfp` Works in the same way as `\acfp`, but starts the long form with an upper case letter.

`\acsp` Works in the same way as `\acs`, but makes the short form into a plural.

`\aclp` Works in the same way as `\acl`, but makes the long form into a plural.

`\Aclp` Works in the same way as `\aclp`, but starts with an upper case letter.

`\acfi` Works in the same way as `\acf`, but prints the Full Name acronym (`\acl`) in italics and the abbreviated form (`\acs`) in upshaped form.

`\Acfi` Works in the same way as `\acfi`, but starts the long form with an upper case letter.

`\acused` Marks an acronym as used, as if it had been called with `\ac`, but without printing anything. This means that in the future only the short form of the acronym will be printed.

`\acsu` Prints the short form of the acronym and marks it as used.

`\aclu` Prints the long form of the acronym and marks it as used.

\Aclu Works in the same way as \aclu, but starts with an upper case letter. Example: \acl{lox}/\acl{lh2} (\acsu{lox}/\acsu{lh2})

\iac Works in the same way as the \ac command but prefixes it with an appropriate indefinite article.

\Iac Works in the same way as the \ac command but prefixes it with an appropriate upper case indefinite article.

\...* The following commands do the same as their unstarred forms, except that the acronym will not be marked as used. If you work with the 'onlyused' option then macros which have only been used with starred commands will not show up. \ac*, \Ac*, \acs*, \acl*, \Acl*, \acf*, \Acf*, \acp*, \Acp*, \acsp*, \aclp*, \Aclp*, \acfp*, \Acfp*, \acfi*, \Acfi*, \acsu*, \aclu*, \Aclu*, \iac* and \Iac*.

2.2 Customization

The appearance of \acs and \acf can be configured in various ways. Of main importance are the package options:

footnote makes the full name of the acronym appear as a footnote.

smaller lets the acronyms appear a bit smaller than the surrounding text. This is in accord with typographic convention. The relsize package is required.

There are three lower-level macros controlling the output. Any acronym printed by \acs is formatted by \acsfont. Similarly, unless the option footnote is specified, \acffont handles the output of \acf, where the included acronym goes through \acfsfont (and \acsfont). The plural and upper case forms are treated accordingly. Usually the three macros do nothing. To give an example, the option smaller makes \acsfont use the command \textsmaller from the relsize package:

\acsfont
\acffont
\acfsfont

```
\renewcommand*{\acsfont}[1]{\textsmaller{#1}}
```

2.3 Defining Acronyms

Acronyms can either defined from an environment specifically introduced for that purpose or by direct definitions.

acronym The acronym environment allows one to define all the acronyms needed by a document at a single place and is self-documenting, since a table of acronyms is automatically produced.

\acro In the acronym environment, acronyms are defined with the command:

\acro{⟨*acronym*⟩}[⟨*short name*⟩]{⟨*full name*⟩}

The first argument ⟨*acronym*⟩ is the acronym string itself and is used in the commands of the previous section such as \ac or \acl, that print the different forms of the acronym.

Because internal commands take ⟨*acronym*⟩ for storing the different forms of the acronym, the TEX code for the acronym is limited by \csname. If the acronym

requires problematic or complicate TEX stuff (font commands, ...), then this code can be given in the optional argument ⟨*short name*⟩. The first argument ⟨*acronym*⟩ is then a simpler string to identify the acronym. For example, an acronym for water can look like this:

```
\acro{H2O}[$\mathrm{H_2O}$]{water}
```

Then `\acs{H2O}` gets "H_2O" and `\acl{H2O}` prints "water".

`\acroextra` Inside the `acronym` environment additional information can be added to the list of acronyms with the `\acroextra` command that will not be included in the normal inline acronyms.

```
\acroextra{⟨additional info⟩}
```

for example:

```
\acro{H2O}[$\mathrm{H_2O}$]
     {Dihydrogen Monoxide\acroextra{ (water)}}
\acro{NA}[\ensuremath{N_{\mathrm A}}]
     {Number of Avogadro\acroextra{ (See \S\protect\ref{A1})}}
```

Note that `\acroextra` must be inserted inside the `\acro` definition and that fragile commands must be protected. Be careful of unnecessary spaces.

The standard format of the acronym list is a `\description` environment. If you pass an optional parameter to the `acronym` environment, the width of the acronym-column will be fitted to the width of the given parameter (which should be the longest acronym). For example, if *HBCI* is the longest acronym used, the list should start with

```
\begin{acronym}[HBCI]
```

`\aclabelfont` The short form of each acronym in the list is formatted using `\aclabelfont`, which typesets its arguments in bold font by default. It can be redefined to produce bold sans-serif labels, for example, with

```
\renewcommand*{\aclabelfont}[1]{\textbf{\textsf{\acsfont{#1}}}}
```

In standard mode, the acronym-list will consist of all defined acronyms, regardless if the the acronym was used in the text before or not. This behavior can be changed by loading the package with the parameter `printonlyused` (used at least once) or `printonlyreused` (use more than once):

```
\usepackage[printonlyused]{acronym}
```

In `printonly(re)used`-mode you can add to each acronym the the page number where it was first used by additionally specifying the option `withpage`.

```
\usepackage[printonlyused,withpage]{acronym}
```

`\newacro` `\acrodef` If one does not want an acronym list to be produced at all, acronyms can be defined directly thanks to the two commands

`\newacro{`⟨*acronym*⟩`}[`⟨*short name*⟩`]{`⟨*full name*⟩`}`
`\acrodef{`⟨*acronym*⟩`}[`⟨*short name*⟩`]{`⟨*full name*⟩`}`

the difference between the two consisting in the fact that the latter makes the acronym definition stored in the `.aux` file. Therefore, the acronym becomes available from start-up in the next run.

Note that all the acronym definitions made by `\acro` in the `acronym` environment are also similarly added to the `.aux` file.

2.3.1 Non standard indefinite articles

Sometimes the indefinite article of an acronym differs between its short form and its long form, for example "a Federal Bureau of Investigation (FBI) agent" and "an FBI agent". To deal with this, the package provides the following three commands

`\newacroindefinite`
`\acrodefindefinite`
`\acroindefinite`

`\acroindefinite{`⟨*acronym*⟩`}{`⟨*short indefinite article*⟩`}{` ⟨*long indefinite article*⟩`}`
`\newacroindefinite{`⟨*acronym*⟩`}{`⟨*short indefinite article*⟩`}{` ⟨*long indefinite article*⟩`}`
`\acrodefindefinite{`⟨*acronym*⟩`}{`⟨*short indefinite article*⟩`}{` ⟨*long indefinite article*⟩`}`

that allow one to define indefinite articles. The `\acroindefinite` command is meant to be used in the `acronym` environment. The difference among the latter two is that `\acrodefindefinite` puts the acronym definition in the `.aux` file, so that the acronym exception is available at the next run from start-up.

When using `\iac` and `\Iac` without first defining an article, the default article is "a".

2.3.2 Non standard and foreign plural forms

When the plural form of an acronym is required, the package typically obtains it as an English plural, by adding an 's'. This happens both for long and short forms. For instance, for an acronym defined as

```
\newacro{IC}{Integrated Circuit}
```

the `\acsp{IC}` command produces "ICs", and the `\aclp{IC}` command produces "Integrated Circuits".

Unfortunately, this is generally not suitable for typesetting in languages different from English, and at times it is not correct even for English. For instance consider the "MP" acronym, commonly used to refer to a "Member of the Parlament". Of course, its long form plural is not "Member of the Parlaments", but "Members of the Parlament". For the short form plural, "MPs" is anyway commonly accepted. The same happens with "SOC (System on a Chip)" or "BUT (Block Under Test)".

In foreign languages, things can be even more complicated. For instance, in Italian, there are different rules for English acronyms used in Italian text and Italian acronyms used in Italian text. The former do not get a plural at all, neither for the long, nor for the short form as in "Un paio di *Integrated Circuit (IC)*". The latter get a plural long form following the natural Italian rules for plurals, and a plural short form that can either be the same as the singular short form, or — at times — a form obtained by doubling those letter of the short form that correspond to words that get a plural in the long form. For instance: "Nucleo Investigativo (NI)" could take a plural as in "Nuclei Investigativi (NNII)", although in modern texts one is more likely to find "Nuclei Investigativi (NI)".

`\acroplural`
`\newacroplural`
`\acrodefplural`

To deal with all these different situations, the package (since version 1.35) has been enriched with the following three commands

`\acroplural{`⟨*acronym*⟩`}[`⟨*short plural*⟩`]{` ⟨*long plural*⟩`}`
`\newacroplural{`⟨*acronym*⟩`}[`⟨*short plural*⟩`]{` ⟨*long plural*⟩`}`
`\acrodefplural{`⟨*acronym*⟩`}[`⟨*short plural*⟩`]{` ⟨*long plural*⟩`}`

that allow one to define plural exceptions. The `\acroplural` command is meant to be used in the `acronym` environment. The difference among the latter two is that `\acrodefplural` puts the acronym definition in the `.aux` file, so that the acronym exception is available at the next run from start-up. When the optional short form is not provided, the acronym name plus an 's' is used.

Plural exceptions are never reported in tables of acronyms.

2.4 Miscellaneous

2.4.1 Sectioning and pdf marks

Acronyms are robust (since version 1.12) and can be used in sectional headers such as `\chapter`, `\section`, etc., but please note the following:

- Do not use the general form (`\ac` or `\acp`) in sectional headers, because it will uses the full name the first time, that is in the table of contents, and the short form further on.
- The text of ⟨*acronym*⟩ is used verbatim in bookmarks and not ⟨*short name*⟩ for pdfTEX with `hyperref`.
- When the long form of the acronym is used in sectional headers (for pdfTEX with `hyperref`), it will end up in the pdf bookmarks. In that case it is good to hide unusual text such as math inside the `\texorpdfstring` defined by `hyperref`, for example:

```
\acro{Nx}[\ensuremath{N_{\chi}}]
     {\texorpdfstring{$\chi$}{X}-factor}
```

  which will then give

pdf bookmark: `\acf{Nx}` → X-factor (Nx)
text: `\acf{Nx}` → χ-factor (N_χ)

- For acronyms in sectional headers, the file must be PDFLaTeX'ed 3 times before the bookmarks are correct.

- Acronyms in sectional headers together with the `footnote` option will not give reliable results, because it will end up in the running heads and table of contents. If you really need it, use the optional argument of the sectioning commands. For example:

```
\chapter[The water \texorpdfstring{$\mathrm{H_2O}$}{H2O}) ...]
        {The \acf{H2O} ...}
```

3 An example file

```
⟨*acrotest⟩
\documentclass{article}
\usepackage[colorlinks]{hyperref}
\usepackage[printonlyused,withpage]{acronym}
\begin{document}

\section{Intro}
In the early nineties, \acs{GSM} was deployed in many European
countries. \ac{GSM} offered for the first time international
roaming for mobile subscribers. The \acs{GSM}'s use of \ac{TDMA} as
its communication standard was debated at length. And every now
and then there are big discussion whether \ac{CDMA} should have
been chosen over \ac{TDMA}.

\section{Furthermore}
\acresetall
The reader could have forgotten all the nice acronyms, so we repeat the
meaning again.

If you want to know more about \acf{GSM}, \acf{TDMA}, \acf{CDMA}
and other acronyms, just read a book about mobile communication. Just
to mention it: There is another \ac{UA}, just for testing purposes!

\begin{figure}[h]
Figure
\caption{A float also admits references like \ac{GSM} or \acf{CDMA}.}
\end{figure}

\subsection{Some chemistry and physics}
\label{Chem}
\ac{NAD+} is a major electron acceptor in the oxidation
of fuel molecules. The reactive part of \ac{NAD+} is its nictinamide
ring, a pyridine derivate.

One mol consists of \acs{NA} atoms or molecules. There is a relation
between the constant of Boltzmann and the \acl{NA}:
\begin{equation}
  k = R/\acs{NA}
\end{equation}

\acl{lox}/\acl{lh2} (\acsu{lox}/\acsu{lh2})

\Acp{LFVP} are processes in which the lepton number of the initial
and final states are different. An example for \iac{LFVP} is
neutrinoless double beta decay.

\subsection{Some testing fundamentals}
When testing \acp{IC}, one typically wants to identify functional
```

```
blocks to be tested separately. The latter are commonly indicated as
\acp{BUT}. To test a \ac{BUT} requires defining a testing strategy\dots{}
\Iac{IC} popped up unexpectedly.

\section{Acronyms}
\begin{acronym}[TDMA]
 \acro{CDMA}{Code Division Multiple Access}
 \acro{GSM}{Global System for Mobile communication}
 \acro{NA}[\ensuremath{N_{\mathrm A}}]
      {Number of Avogadro\acroextra{ (see \S\ref{Chem})}}
 \acro{NAD+}[NAD\textsuperscript{+}]{Nicotinamide Adenine Dinucleotide}
 \acro{LFVP}{lepton flavor violating process}
 \acroindefinite{LFVP}{an}{a}
 \acro{NUA}{Not Used Acronym}
 \acro{TDMA}{Time Division Multiple Access}
 \acro{UA}{Used Acronym}
 \acro{lox}[\ensuremath{LOX}]{Liquid Oxygen}%
 \acro{lh2}[\ensuremath{LH_2}]{Liquid Hydrogen}%
 \acro{IC}{Integrated Circuit}%
 \acro{BUT}{Block Under Test}%
 \acrodefplural{BUT}{Blocks Under Test}%
 \acroindefinite{IC}{an}{an}
\end{acronym}

\end{document}
⟨/acrotest⟩
```

4 The implementation

⟨*acronym⟩

4.1 Identification

First we test that we got the right format and name the package.

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesPackage{acronym}[2020/04/17
                          v1.47
                          Support for acronyms (Tobias Oetiker)]
\RequirePackage{suffix,xstring}
```

4.2 Options

\ifAC@footnote The option `footnote` leads to a redefinition of `\acf`, `\Acf`, `\acfp`, and `\Acfp`, making the full name appear as a footnote.

```
\newif\ifAC@footnote
\AC@footnotefalse
\DeclareOption{footnote}{\AC@footnotetrue}
```

\ifAC@nohyperlinks If hyperref is loaded, all acronyms will link to their glossary entry. With the option `nohyperlinks` these links can be suppressed.

```
\newif\ifAC@nohyperlinks
\AC@nohyperlinksfalse
\DeclareOption{nohyperlinks}{\AC@nohyperlinkstrue}
```

\ifAC@noacroprefix With the `noacroprefix` option the acronym commands are not prefixed. This reproduces the old behavior of version ¡1.43, but can cause collisions between user-defined acronyms and commands of this package.

```
\newif\ifAC@noacroprefix
\AC@noacroprefixfalse
\DeclareOption{noacroprefix}{\AC@noacroprefixtrue}
```

\ifAC@printonlyused We need a marker which is set if the option `printonlyused` was used.

```
\newif\ifAC@printonlyused
\AC@printonlyusedfalse
\DeclareOption{printonlyused}{\AC@printonlyusedtrue}
```

\ifAC@printonlyreused With the `printonlyreused` option, only those acronyms are included in the list of acronyms that have been used more than once, i.e. at least twice.

```
\newif\ifAC@printonlyreused
\AC@printonlyreusedfalse
\DeclareOption{printonlyreused}{\AC@printonlyreusedtrue}
```

\ifAC@withpage A marker which tells us to print page numbers.

```
\newif\ifAC@withpage
\AC@withpagefalse
\DeclareOption{withpage}{\AC@withpagetrue}
```

\ifAC@smaller The option `smaller` leads to a redefinition of `\acsfont`. We want to make the acronym appear smaller. Since this should be done in a context-sensitive way, we rely on the macro `\textsmaller` provided by the `relsize` package. As `\RequirePackage` cannot be used inside `\DeclareOption`, we need a boolean variable.

```
\newif\ifAC@smaller
\AC@smallerfalse
\DeclareOption{smaller}{\AC@smallertrue}
```

\ifAC@dua The option `dua` stands for "don't use acronyms". It leads to a redefinition of `\ac`, `\Ac`, `\acp`, and `\Acp`, making the full name appear all the time and suppressing all acronyms but the explicity requested by `\acf`, `\Acf`, `\acfp` or `\Acfp`.

```
\newif\ifAC@dua
\AC@duafalse
\DeclareOption{dua}{\AC@duatrue}
```

\ifAC@nolist The option `nolist` stands for "don't write the list of acronyms".

```
\newif\ifAC@nolist
\AC@nolistfalse
\DeclareOption{nolist}{\AC@nolisttrue\AC@nohyperlinkstrue}
```

\ifAC@nolinebreak The option `nolinebreak` dictates whether to forbid, by defalt, a line break between the full name and the short name, when they are presented together.

```
\newif\ifAC@nolinebreak
\AC@nolinebreakfalse
\DeclareOption{nolinebreak}{\AC@nolinebreaktrue}
```

Now we process the options.

```
\ProcessOptions\relax
```

4.3 Setup macros

\acsfont \acffont \acfsfont The appearance of the output of the commands `\acs` and `\acf` is partially controlled by `\acsfont`, `\acffont`, and `\acfsfont`. By default, they do nothing except when the `smaller` option is loaded.

The option `smaller` leads to a redefinition of `\acsfont`. We want to make the acronym appear smaller. Since this should be done in a context-sensitive way, we rely on the macro `\textsmaller` provided by the `relsize` package.

```
\ifAC@smaller
  \RequirePackage{relsize}
  \newcommand*{\acsfont}[1]{\textsmaller{#1}}
\else
```

```
  \newcommand*{\acsfont}[1]{#1}
\fi
\newcommand*{\acffont}[1]{#1}
\newcommand*{\acfsfont}[1]{#1}
```

`\AC@linebreakpenalty` When the option `nolinebreak` is specified, the default penalty for a line break is being set to the maximum. Otherwise, the default penalty is one level below the maximum, meaning that most of the times, by default, the line will not get broken.

```
\ifAC@nolinebreak
  \def\AC@linebreakpenalty{4}
\else
  \def\AC@linebreakpenalty{3}
\fi
```

4.4 Hyperlinks and PDF support

`\AC@hyperlink` `\AC@hyperref` `\AC@hypertarget` `\AC@phantomsection` Define dummy hyperlink commands

```
\def\AC@hyperlink#1#2{#2}
\def\AC@hyperref[#1]#2{#2}
\def\AC@hypertarget#1#2{#2}
\def\AC@phantomsection{}
```

`\AC@raisedhypertarget` Make sure that hyperlink processing gets enabled before we process the document if hyperref has been loaded in the mean time.

```
\ifAC@nohyperlinks
\else
   \AtBeginDocument{%
      \@ifpackageloaded{hyperref}
         {\let\AC@hyperlink=\hyperlink
          \let\AC@hyperref=\hyperref
          \newcommand*\AC@raisedhypertarget[2]{%
             \Hy@raisedlink{\hypertarget{#1}{}}#2}%
          \let\AC@hypertarget=\AC@raisedhypertarget
          \def\AC@phantomsection{%
            \Hy@GlobalStepCount\Hy@linkcounter
            \edef\@currentHref{section*.\the \Hy@linkcounter}%
            \Hy@raisedlink{%
              \hyper@anchorstart{\@currentHref}\hyper@anchorend
            }%
          }%
         }{}}%
\fi
```

`\AC@pageref` Use `\pageref*` instead of `\pageref` when the `hyperref` package is used.

```
\AtBeginDocument{%
  \@ifpackageloaded{hyperref}{%
    \let\AC@pageref=\@pagerefstar%
```

```
  }{%
    \let\AC@pageref=\pageref%
  }%
}
```

The `hyperref` package defines `\pdfstringdefDisableCommands` and `\texorpdfstring` for text in bookmarks. If undefined, then provide them it at the beginning of the document.

```
\AtBeginDocument{%
   \providecommand\texorpdfstring[2]{#1}%
   \providecommand\pdfstringdefDisableCommands[1]{}%
   \pdfstringdefDisableCommands{%
     \csname AC@starredfalse\endcsname
     \csname AC@footnotefalse\endcsname
     \let\AC@hyperlink\@secondoftwo
     \let\acsfont\relax
     \let\acffont\relax
     \let\acfsfont\relax
     \let\acused\relax
     \let\null\relax
     \def\AChy@call#1#2{%
        \ifx*#1\@empty
          \expandafter #2%
        \else
          #2{#1}%
        \fi
      }%
      \def\acs#1{\AChy@call{#1}\AC@acs}%
      \def\acl#1{\AChy@call{#1}\@acl}%
      \def\Acl#1{\AChy@call{#1}\@Acl}%
      \def\acf#1{\AChy@call{#1}\AChy@acf}%
      \def\Acf#1{\AChy@call{#1}\AChy@Acf}%
      \def\ac#1{\AChy@call{#1}\@ac}%
      \def\Ac#1{\AChy@call{#1}\@Ac}%
      \def\acsp#1{\AChy@call{#1}\@acsp}%
      \def\aclp#1{\AChy@call{#1}\@aclp}%
      \def\Aclp#1{\AChy@call{#1}\@Aclp}%
      \def\acfp#1{\AChy@call{#1}\AChy@acfp}%
      \def\Acfp#1{\AChy@call{#1}\AChy@Acfp}%
      \def\acp#1{\AChy@call{#1}\@acp}%
      \def\Acp#1{\AChy@call{#1}\@Acp}%
      \def\acfi#1{\AChy@call{#1}\AChy@acf}%
      \def\Acfi#1{\AChy@call{#1}\AChy@Acf}%
      \let\acsu\acs
      \let\aclu\acl
      \let\Aclu\Acl
      \def\AChy@acf#1{\AC@acl{#1} (\AC@acs{#1})}%
      \def\AChy@Acf#1{\AC@Acl{#1} (\AC@acs{#1})}%
      \def\AChy@acfp#1{\AC@aclp{#1} (\AC@acsp{#1})}%
```

```
      \def\AChy@Acfp#1{\AC@Aclp{#1} (\AC@acsp{#1})}%
   }%
}
```

4.5 Additional Helper macros

We need a list of the used acronyms after the last `\acresetall` (or since beginning), a token list is very useful for this purpose

`AC@clearlist`

```
\newtoks\AC@clearlist
```

`\AC@addtoAC@clearlist` Adds acronyms to the clear list

```
\newcommand*\AC@addtoAC@clearlist[1]{%
  \global\AC@clearlist\expandafter{\the\AC@clearlist\AC@reset{#1}}%
}
```

`\acresetall` `\AC@reset` This macro resets the `AC@FN` - tag of each acronym, therefore `\ac` will use Full Name (FN) next time it is called

```
\newcommand*\acresetall{\the\AC@clearlist\AC@clearlist={}}
\def\AC@reset#1{%
  \global\expandafter\let\csname AC@\AC@prefix#1\endcsname\relax
}
```

`\AC@used` We also need a markers for 'used'.

```
\newcommand*\AC@used{@<>@<>@}
```

`\AC@populated` An on/off flag to note if any acronyms were logged. This is needed for the first run with `printonly(re)used` option, because the acronym list are then empty, resulting in a `"missing item"` error.

```
\newcommand{\AC@populated}{}
```

`\AC@logged` `\acronymused` Log the usage by writing the `\acronymused` to the `aux` file and by reading it back again at the beginning of the document (performed automatically by LaTeX). This results in processing the document twice, but it is needed anyway for the rest of the package.

This methodology is needed when the list of acronyms is in the front matter of the document.

```
\newcommand*{\AC@logged}[1]{%
   \@bsphack
   \protected@write\@auxout{}{\string\acronymused{#1}}%
   \@esphack}
```

Keep it out of bookmarks.

```
\AtBeginDocument{%
   \pdfstringdefDisableCommands{%
      \let\AC@logged\@gobble
   }%
}
```

Flag the acronym at the beginning of the document as used (called by the `aux` file).

```
\newcommand*{\acronymused}[1]{%
  \expandafter\ifx\csname acused@#1@once\endcsname\AC@used%
    \expandafter\ifx\csname acused@#1@twice\endcsname\AC@used%
      \relax%
    \else%
      \global\expandafter\let\csname acused@#1@twice\endcsname\AC@used%
      \global\let\AC@populated\AC@used%
    \fi%
  \else%
    \global\expandafter\let\csname acused@#1@once\endcsname\AC@used%
    \ifAC@printonlyreused%
      \relax%
    \else%
      \global\let\AC@populated\AC@used%
    \fi%
  \fi%
}
```

`\@firstupper` Internal commands for making a first letter upper case.

```
\newcommand{\@firstupper}[1]{%
  \StrSplit{#1}{1}{\head}{\tail}%
  \MakeUppercase\head\tail%
}
```

`AC@prefix` Returns the prefix used to build the defined acronym commands as long as the `noacroprefix` option is disabled. Otherwise the output is empty, so the old behaviour from version ¡1.43 is reproduced.

```
\ifAC@noacroprefix
  \newcommand*\AC@prefix{}
\else
  \newcommand*\AC@prefix{acronyms@}
\fi
```

4.6 Defining acronyms

There are three commands that define acronyms: `\newacro`, `\acrodef`, and `\acro`. They are called with the following arguments:

`\acro{`⟨*acronym*⟩`}[`⟨*short name*⟩`]{`⟨*full name*⟩`}`

The mechanism used in this package is to make the optional ⟨*short name*⟩ identical to the ⟨*acronym*⟩ when it is empty (no optional argument), thereby only the second (optional) argument is stored together with the ⟨*full name*⟩.

`\newacro` `\AC@newacro` The internal macro `\newacro` stores the ⟨*short name*⟩ and the ⟨*full name*⟩ in the command `\fn@<`*acronym*`>`.

```
\newcommand*\newacro[1]{%
  \@ifnextchar[{\AC@newacro{#1}}{\AC@newacro{#1}[#1]}}
```

```
\newcommand\AC@newacro{}
\def\AC@newacro#1[#2]#3{%
   \expandafter\gdef\csname fn@#1\endcsname{{#2}{#3}}%
   }
```

`\acrodef` `\AC@acrodef` The user command `\acrodef` calls `\newacro` and writes it into the `.aux` file.

```
\newcommand*\acrodef[1]{%
  \@ifnextchar[{\AC@acrodef{#1}}{\AC@acrodef{#1}[#1]}}

\newcommand\AC@acrodef{}
\def\AC@acrodef#1[#2]#3{%
   \@bsphack
   \protected@write\@auxout{}{\string\newacro{#1}[#2]{#3}}%
   \@esphack}
```

`AC@deflist` In standard mode, the acronym - list is formatted with a description environment. If an optional argument is passed to the acronym environment, the list is formatted as a AC@deflist, which needs the longest appearing acronym as parameter. If the option 'nolist' is selected the enviroment is empty.

```
\newcommand*{\aclabelfont}[1]{\textbf{\acsfont{#1}}}
\def\AC@makelabel#1{#1\hfil}
\newenvironment{AC@deflist}[1]%
        {\ifAC@nolist%
         \else%
            \raggedright\begin{list}{}%
                {\settowidth{\labelwidth}{\AC@makelabel{\aclabelfont{#1}}}%
                \setlength{\leftmargin}{\labelwidth}%
                \addtolength{\leftmargin}{\labelsep}%
                \renewcommand{\makelabel}{\AC@makelabel}}%
          \fi}%
        {\ifAC@nolist%
         \else%
            \end{list}%
         \fi}%
```

`acronym` In the 'acronym' - environment, all acronyms are defined, and printed if they have been used before, which is indicated by the acused-tag.

```
\begin{acronym}
\acro{CDMA}{Code Division Multiple Access\acroextra{\ ...}}
\end{acronym}
```

`\acroextra` Additional information can be added after to `\acro` definition for display in the list of acronyms. This command is only active inside the `acronym` environment. Outside it gobbles up its argument.

```
\newcommand{\acroextra}[1]{}
```

`\acro` Acronyms can be defined with the user command `\acro` in side the `acronym` environment.

```
\newenvironment{acronym}[1][1]{%
   \providecommand*{\acro}{\AC@acro}%
   \providecommand*{\acroplural}{\AC@acroplural}%
   \providecommand*{\acroindefinite}{\AC@acroindefinite}%
   \long\def\acroextra##1{##1}%
   \def\@tempa{1}\def\@tempb{#1}%
   \ifx\@tempa\@tempb%
      \global\expandafter\let\csname AC@des@mark\endcsname\AC@used%
      \ifAC@nolist%
      \else%
         \begin{description}%
      \fi%
   \else%
      \begin{AC@deflist}{#1}%
   \fi%
  }%
  {%
   \ifx\AC@populated\AC@used\else%
      \ifAC@nolist%
      \else%
          \item[]\relax%
      \fi%
   \fi%
   \expandafter\ifx\csname AC@des@mark\endcsname\AC@used%
      \ifAC@nolist%
      \else%
        \end{description}%
      \fi%
   \else%
      \end{AC@deflist}%
   \fi}%
```

\AC@acro
\AC@@acro

```
\newcommand*\AC@acro[1]{%
  \@ifnextchar[{%
    \csname AC@\AC@prefix{}@acro\endcsname{#1}%
  }{%
    \csname AC@\AC@prefix{}@acro\endcsname{#1}[#1]%
  }%
}
\expandafter\newcommand\csname AC@\AC@prefix{}@acro\endcsname{}
\expandafter\def\csname AC@\AC@prefix{}@acro\endcsname#1[#2]#3{%
  \ifAC@nolist%
  \else%
  \ifnum%
    \ifAC@printonlyused 1%
    \else\ifAC@printonlyreused 1%
    \else 0\fi\fi%
  =1\relax%
```

```
  \ifnum%
    \ifAC@printonlyused%
      \expandafter\ifx\csname acused@#1@once\endcsname\AC@used 1 \else 0 \fi%
    \else\ifAC@printonlyreused%
      \expandafter\ifx\csname acused@#1@twice\endcsname\AC@used 1 \else 0 \fi%
    \else 0 \fi\fi%
  =1\relax%
    \item[\protect\AC@hypertarget{#1}{%
      \AC@hyperref[acro:#1]{\aclabelfont{#2}\hfill}%
    }]\AC@hyperref[acro:#1]{#3}%
        \ifAC@withpage%
          \expandafter\ifx\csname r@acro:#1\endcsname\relax%
             \PackageInfo{acronym}{%
               Acronym #1 used in text but not spelled out in
               full in text}%
          \else%
            \nobreak\leaders\hbox{$\m@th\mkern\@dotsep mu\hbox{.}\mkern\@dotsep mu$}\hfill%
            \nobreak\hb@xt@\@pnumwidth{%
              \hfil\normalfont\normalcolor\AC@pageref{acro:#1}%
            }%
          \fi%
        \fi\\%
  \fi%
\else%
  \item[\protect\AC@hypertarget{#1}{\AC@hyperref[acro:#1]{\aclabelfont{#2}\hfill}}]\AC@hyperr
\fi%
\fi%
\begingroup
  \def\acroextra##1{}%
  \@bsphack
    \ifAC@printonlyreused%
      \protected@write\@auxout{}{%
        \string\newacro{#1}[%
          \expandafter\ifx\csname acused@#1@twice\endcsname\AC@used%
            \string\AC@hyperlink{#1}{#2}%
          \else%
            {#2}%
          \fi%
        ]{#3}%
      }%
    \else%
      \protected@write\@auxout{}{%
        \string\newacro{#1}[\string\AC@hyperlink{#1}{#2}]{#3}%
      }%
    \fi%
  \@esphack
\endgroup
\ignorespaces}
```

4.6.1 Nonstandard indefinite articles

`\newacroindefinite` Sets up a non standard indefinite article for a given acronym.

```
\newcommand*\newacroindefinite[3]{%
  \expandafter\gdef\csname fn@#1@IS\endcsname{#2}%
  \expandafter\gdef\csname fn@#1@IL\endcsname{#3}%
}
```

`\acrodefindefinite` Same as above, storing content in aux file.

```
\newcommand*\acrodefindefinite[3]{%
  \@bsphack
  \protected@write\@auxout{}{\string\newacroindefinite{#1}{#2}{#3}}%
  \@esphack
}
```

`\AC@acroindefinite` Internal command to set up an indefinite article in the acronym environment.

```
\newcommand\AC@acroindefinite[3]{
  \@bsphack
  \protected@write\@auxout{}%
    {\string\newacroindefinite{#1}{#2}{#3}}%
  \@esphack
}
```

4.6.2 Non standard or foreign plural forms

`\newacroplural` `\AC@newacroplurali` `\AC@newacropluralii` Sets up a non standard plural form for a given acronym.

```
\newcommand*\newacroplural[1]{%
  \@ifnextchar[%]
  {\AC@newacroplurali{#1}}{\AC@newacropluralii{#1}}%
}
\newcommand\AC@newacroplurali{}
\def\AC@newacroplurali#1[#2]#3{%
  \expandafter\gdef\csname fn@#1@PS\endcsname{#2}%
  \expandafter\gdef\csname fn@#1@PL\endcsname{#3}%
}
\newcommand\AC@newacropluralii[2]{%
  \expandafter\gdef\csname fn@#1@PL\endcsname{#2}%
}
```

`\acrodefplural` `\AC@acrodefplurali` `\AC@acrodefpluralii` Same as above, storing content in aux file.

```
\newcommand*\acrodefplural[1]{%
   \@ifnextchar[%]
   {\AC@acrodefplurali{#1}}{\AC@acrodefpluralii{#1}}%
}
\newcommand\AC@acrodefplurali{}
\def\AC@acrodefplurali#1[#2]#3{%
  \@bsphack
  \protected@write\@auxout{}{\string\newacroplural{#1}[#2]{#3}}%
  \@esphack
```

```
}
\newcommand\AC@acrodefpluralii[2]{%
  \@bsphack
  \protected@write\@auxout{}{\string\newacroplural{#1}{#2}}%
  \@esphack
}
```

\AC@acroplural
\AC@acroplurali
\AC@acropluralii

Internal commands to set up a plural version of an acronym in the acronym environment.

```
\newcommand*\AC@acroplural[1]{%
   \@ifnextchar[%]
   {\AC@acroplurali{#1}}{\AC@acropluralii{#1}}%
}
\newcommand\AC@acroplurali{}
\def\AC@acroplurali#1[#2]#3{%
  \@bsphack
  \protected@write\@auxout{}%
    {\string\newacroplural{#1}[\string\AC@hyperlink{#1}{#2}]{#3}}%
  \@esphack
}
\newcommand\AC@acropluralii[2]{
  \@bsphack
  \protected@write\@auxout{}%
    {\string\newacroplural{#1}[\string\AC@hyperlink{#1}{\AC@acs{#1}}]{#2}}%
  \@esphack
}
```

\AC@aclp
\AC@Aclp
\AC@acsp

Deliver either standard or nonstandard plural form (long and short respectively).

```
\newcommand*\AC@aclp[1]{%
  \ifcsname fn@#1@PL\endcsname
  \csname fn@#1@PL\endcsname
  \else
  \AC@acl{#1}s%
  \fi
}
\newcommand*\AC@Aclp[1]{%
  \AC@uppertrue%
  \AC@aclp{#1}%
  \AC@upperfalse%
}
\newcommand*\AC@acsp[1]{%
  \ifcsname fn@#1@PS\endcsname
  \csname fn@#1@PS\endcsname
  \else
  \AC@acs{#1}s%
  \fi
}
```

4.7 Using acronyms

\ifAC@starred Before the macros are defined, we need a boolean variable which will be set to true or false, when the following commands are used in the starred or unstarred form. If it is true, the acronym will be not be logged, otherwhise it will be logged.

```
\newif\ifAC@starred
```

\ifAC@upper If an acronym needs to be capitalized, this flag is used to indicate this at an appropriate point in the code. In that case, the firstupper command will be called at a time when the acronym is expandable, otherwise the xstring command will not work properly.

```
\newif\ifAC@upper
```

\AC@get If the acronym is undefined, the internal macro \AC@get warns the user by printing the name in bold with an exclamation mark at the end. If defined, \AC@get uses the same mechanism used by the LaTeX kernel commands \ref and \pageref to return the short \AC@acs and long forms \AC@acl of the acronym saved in \fn@<*acronym*>.

```
\newcommand*\AC@get[3]{%
  \ifx#1\relax
    \PackageWarning{acronym}{Acronym `#3' is not defined}%
    \textbf{#3!}%
  \else
    \ifAC@upper
      \@firstupper{\expandafter#2#1}%
    \else
      \expandafter#2#1%
    \fi
  \fi
}
```

\AC@acs \AC@acl \AC@Acl The internal commands \AC@acs and \AC@acl returns the (unformatted) short and the long forms of an acronym as saved in \fn@<*acronym*>. Mbox to prevent hyphenation of short form.

```
\newcommand*\AC@acs[1]{%
   \mbox{\expandafter\AC@get\csname fn@#1\endcsname\@firstoftwo{#1}}}
\newcommand*\AC@acl[1]{%
   \expandafter\AC@get\csname fn@#1\endcsname\@secondoftwo{#1}}
\newcommand*\AC@Acl[1]{%
  \AC@uppertrue%
  \AC@acl{#1}%
  \AC@upperfalse%
}
```

\acs \acsa \@acs The user macro \acs prints the short form of the acronym using the font specified by \acsfont.

```
\newcommand*{\acs}{\AC@starredfalse\protect\acsa}%
\WithSuffix\newcommand\acs*{\AC@starredtrue\protect\acsa}%
```

```
\newcommand*{\acsa}[1]{%
   \texorpdfstring{\protect\@acs{#1}}{#1}}
\newcommand*{\@acs}[1]{%
   \acsfont{\AC@acs{#1}}%
%% having a footnote on acs sort of defeats the purpose
%%   \ifAC@footnote
%%      \footnote{\AC@acl{#1}{}}%
%%   \fi
   \ifAC@starred\else\AC@logged{#1}\fi}
```

\acl The user macro \acl prints the full name of the acronym.

\@acl \Acl \@Acl

```
\newcommand*{\acl}{\AC@starredfalse\protect\@acl}%
\WithSuffix\newcommand\acl*{\AC@starredtrue\protect\@acl}%
\newcommand*{\Acl}{\AC@starredfalse\protect\@Acl}%
\WithSuffix\newcommand\Acl*{\AC@starredtrue\protect\@Acl}%
\newcommand*{\@acl}[1]{%
   \AC@acl{#1}%
   \ifAC@starred\else\AC@logged{#1}\fi}
\newcommand*{\@Acl}[1]{%
   \AC@Acl{#1}%
   \ifAC@starred\else\AC@logged{#1}\fi}
```

4.8 Helper functions to unset labels

\@verridelabel The internal \@verridelabel command lets us 'redefine' an acronym label such that the page reference in the acronym list points where it should be pointing and not just to the very first occurrence of the acronym, where it may not even be expanded. (code by Ulrich Diez)

```
\newcommand*\@verridelabel[1]{%
  \@bsphack
  \protected@write\@auxout{}{\string\AC@undonewlabel{#1}}%
  \label{#1}%
  \AC@overriddenmessage rs{#1}%
  \@esphack
}%
\newcommand*\AC@undonewlabel{\AC@und@newl@bel rs}%
\newcommand*\AC@und@newl@bel[3]{%
  \@ifundefined{#1@#3}%
  {%
    \global\expandafter\let\csname#2@#3\endcsname\@nnil
  }%
  {%
    \global\expandafter\let\csname#1@#3\endcsname\relax
  }%
}%
\newcommand*\AC@overriddenmessage[3]{%
  \expandafter\ifx\csname#2@#3\endcsname\@nnil
```

```
    \expandafter\@firstoftwo
  \else
    \@ifundefined{#1@#3}%
    {%
      \@ifundefined{#2@#3}%
      {\expandafter\@firstoftwo}%
      {\expandafter\@secondoftwo}%
    }%
    {\expandafter\@secondoftwo}%
  \fi
  {%
    \PackageInfo{acronym}{Label `#3' newly defined as it
    shall be overridden^^Jalthough it is yet undefined}%
    \global\expandafter\let\csname#2@#3\endcsname\empty
  }%
  {%
    \PackageInfo{acronym}{Label `#3' overridden}%
    \@ifundefined{#2@#3}{%
      \global\expandafter\let\csname#2@#3\endcsname\empty}{}%
    \expandafter\g@addto@macro\csname#2@#3\endcsname{i}%
  }%
}%
\newcommand*\AC@testdef[3]{%
  \@ifundefined{s@#2}\@secondoftwo\@firstofone
  {%
    \expandafter\ifx\csname s@#2\endcsname\empty
      \expandafter\@firstofone
    \else
      \expandafter\xdef\csname s@#2\endcsname{%
        \expandafter\expandafter
        \expandafter\@gobble
        \csname s@#2\endcsname
      }%
      \expandafter\@gobble
    \fi
  }%
  {%
    \@testdef{#1}{#2}{#3}%
  }%
}%
\AtBeginDocument{\immediate\write\@auxout{\string\AC@reset@newl@bel}}
\newcommand*\AC@reset@newl@bel{%
  \ifx\@newl@bel\@testdef
    \let\@newl@bel\AC@testdef
    \let\AC@undonewlabel\@gobble
  \fi
}%
\newcommand*\AC@placelabel[1]{%
  \expandafter\ifx\csname AC@\AC@prefix#1\endcsname\AC@used
  \else
```

```
    {\AC@phantomsection\@verridelabel{acro:#1}}%
    \ifAC@starred\else%
    \global\expandafter\let\csname AC@\AC@prefix#1\endcsname\AC@used
    \fi%
    \AC@addtoAC@clearlist{#1}%
  \fi
}%
```

\acf
\acfa
\@acf
\Acf
\Acfa
\@Acf

The user macro \acf always prints the full name with the acronym. The format depends on \acffont and \acfsfont, and on the option footnote handled below. The acronym is added to the clear list to keep track of the used acronyms and it is marked as used by by \gdefining the \AC@FN to be \AC@used after its first use.

The option footnote leads to a redefinition of \acf, making the full name appear as a footnote. There is then no need for \acffont and \acfsfont. If the option footnote is not specified, the optional variable determines the penalty for a line break.

```
\newcommand*{\acf}{\AC@starredfalse\protect\acfa}%
\WithSuffix\newcommand\acf*{\AC@starredtrue\protect\acfa}%
\newcommand*{\Acf}{\AC@starredfalse\protect\Acfa}%
\WithSuffix\newcommand\Acf*{\AC@starredtrue\protect\Acfa}%
\newcommand*{\acfa}[2][\AC@linebreakpenalty]{%
  \texorpdfstring{\protect\@acf[#1]{#2}}{\AC@acl{#2} (#2)}}
\newcommand*{\Acfa}[2][\AC@linebreakpenalty]{%
  \texorpdfstring{\protect\@Acf[#1]{#2}}{\AC@Acl{#2} (#2)}}
\newcommand*{\@acf}[2][\AC@linebreakpenalty]{%
   \ifAC@footnote
      \acsfont{\AC@acs{#2}}%
      \footnote{\AC@placelabel{#2}\AC@acl{#2}{}}%
   \else
      \acffont{%
         \AC@placelabel{#2}\AC@acl{#2}%
         \nolinebreak[#1] %
         \acfsfont{(\acsfont{\AC@acs{#2}})}%
       }%
    \fi
    \ifAC@starred\else\AC@logged{#2}\fi}
\newcommand*{\@Acf}[2][\AC@linebreakpenalty]{%
   \ifAC@footnote
      \acsfont{\AC@acs{#2}}%
      \footnote{\AC@placelabel{#2}\AC@Acl{#2}{}}%
   \else
      \acffont{%
         \AC@placelabel{#2}\AC@Acl{#2}%
         \nolinebreak[#1] %
         \acfsfont{(\acsfont{\AC@acs{#2}})}%
       }%
    \fi
    \ifAC@starred\else\AC@logged{#2}\fi}
```

\ac
\@ac
\Ac
\@Ac

The first time an acronym is accessed its Full Name (FN) is printed. The next time just (FN). When the `footnote` option is used the short form (FN) is always used. The optional variable is being passed to `\acf`, in case it is used.

```
\newcommand*{\ac}{\AC@starredfalse\protect\@ac}%
\WithSuffix\newcommand\ac*{\AC@starredtrue\protect\@ac}%

\newcommand*{\Ac}{\AC@starredfalse\protect\@Ac}%
\WithSuffix\newcommand\Ac*{\AC@starredtrue\protect\@Ac}%

\newcommand{\@ac}[2][\AC@linebreakpenalty]{%
  \ifAC@dua
     \ifAC@starred\acl*{#2}\else\acl{#2}\fi%
  \else
     \expandafter\ifx\csname AC@\AC@prefix#2\endcsname\AC@used%
     \ifAC@starred\acs*{#2}\else\acs{#2}\fi%
   \else
     \ifAC@starred\acf*[#1]{#2}\else\acf[#1]{#2}\fi%
   \fi
  \fi}

\newcommand{\@Ac}[2][\AC@linebreakpenalty]{%
  \ifAC@dua
     \ifAC@starred\Acl*{#2}\else\Acl{#2}\fi%
  \else
     \expandafter\ifx\csname AC@\AC@prefix#2\endcsname\AC@used%
     \ifAC@starred\acs*{#2}\else\acs{#2}\fi%
   \else
     \ifAC@starred\Acf*[#1]{#2}\else\Acf[#1]{#2}\fi%
   \fi
  \fi}
```

\iac
\@iac
\@iaci
\Iac
\@Iac

Indefinite article correct expansion. The optional variable is being passed to `\ac`.

```
\newcommand*{\iac}{\AC@starredfalse\protect\@iac}%
\WithSuffix\newcommand\iac*{\AC@starredtrue\protect\@iac}%
\newcommand*{\Iac}{\AC@starredfalse\protect\@Iac}%
\WithSuffix\newcommand\Iac*{\AC@starredtrue\protect\@Iac}%

\newcommand*{\@iaci}[1]{%
  \ifcsname fn@#1@IL\endcsname
    \ifAC@dua
      \csname fn@#1@IL\endcsname%
    \else
      \expandafter\ifx\csname AC@\AC@prefix#1\endcsname\AC@used%
        \csname fn@#1@IS\endcsname%
      \else
        \csname fn@#1@IL\endcsname%
      \fi
    \fi
  \else
    a%
  \fi
}
```

```
\newcommand*{\@iac}[2][\AC@linebreakpenalty]{%
   \@iaci{#2} \ifAC@starred\ac*[#1]{#2}\else\ac[#1]{#2}\fi%
}
\newcommand*{\@Iac}[2][\AC@linebreakpenalty]{%
  \@firstupper{\@iaci{#2}}\space%
  \ifAC@starred\ac*[#1]{#2}\else\ac[#1]{#2}\fi%
}
```

\acsp
\acspa
\@acsp

The user macro \acsp prints the plural short form of the acronym. This is the acronym itself or the ⟨*short name*⟩, if the optional argument is given in the definition of the acronym plus an 's'.

```
\newcommand*{\acsp}{\AC@starredfalse\protect\acspa}%
\WithSuffix\newcommand\acsp*{\AC@starredtrue\protect\acspa}%

\newcommand*{\acspa}[1]{%
   \texorpdfstring{\protect\@acsp{#1}}{\AC@acsp{#1}}}

\newcommand*{\@acsp}[1]{%
   \acsfont{\AC@acsp{#1}}%
   \ifAC@starred\else\AC@logged{#1}\fi}
```

\aclp
\@aclp
\Aclp
\@Aclp

The user macro \aclp prints the plural full name of the acronym.

```
\newcommand*{\aclp}{\AC@starredfalse\protect\@aclp}%
\WithSuffix\newcommand\aclp*{\AC@starredtrue\protect\@aclp}%

\newcommand*{\Aclp}{\AC@starredfalse\protect\@Aclp}%
\WithSuffix\newcommand\Aclp*{\AC@starredtrue\protect\@Aclp}%

\newcommand*{\@aclp}[1]{%
   \AC@aclp{#1}%
   \ifAC@starred\else\AC@logged{#1}\fi}

\newcommand*{\@Aclp}[1]{%
   \AC@Aclp{#1}%
   \ifAC@starred\else\AC@logged{#1}\fi}
```

\acfp
\acfpa
\@acfp
\Acfp
\Acfpa
\@Acfp

The user macro \acfp always prints the plural full name with the plural of the acronym. The format depends on \acffont and \acfsfont, and on the option footnote handled below.

The option footnote leads to a redefinition of \acfp, making the full name appear as a footnote. There is then no need for \acffont and \acfsfont. If the option footnote is not specified, the optional variable determines the penalty for a line break.

```
\newcommand*{\acfp}{\AC@starredfalse\protect\acfpa}%
\WithSuffix\newcommand\acfp*{\AC@starredtrue\protect\acfpa}%

\newcommand*{\Acfp}{\AC@starredfalse\protect\Acfpa}%
\WithSuffix\newcommand\Acfp*{\AC@starredtrue\protect\Acfpa}%

\newcommand*{\acfpa}[2][\AC@linebreakpenalty]{%
   \texorpdfstring{\protect\@acfp[#1]{#2}}{\AC@aclp{#2} (\AC@acsp{#2})}}

\newcommand*{\Acfpa}[2][\AC@linebreakpenalty]{%
   \texorpdfstring{\protect\@Acfp[#1]{#2}}{\AC@Aclp{#2} (\AC@acsp{#2})}}
```

```
\newcommand*{\@acfp}[2][\AC@linebreakpenalty]{%
   \ifAC@footnote
      \acsfont{\AC@acsp{#2}}%
      \footnote{\AC@placelabel{#2}\AC@aclp{#2}{}}%
   \else
      \acffont{%
         \AC@placelabel{#2}\AC@aclp{#2}%
         \nolinebreak[#1] %
         \acfsfont{(\acsfont{\AC@acsp{#2}})}%
         }%
   \fi
   \ifAC@starred\else\AC@logged{#2}\fi}

\newcommand*{\@Acfp}[2][\AC@linebreakpenalty]{%
   \ifAC@footnote
      \acsfont{\AC@acsp{#2}}%
      \footnote{\AC@placelabel{#2}\AC@Aclp{#2}{}}%
   \else
      \acffont{%
         \AC@placelabel{#2}\AC@Aclp{#2}%
         \nolinebreak[#1] %
         \acfsfont{(\acsfont{\AC@acsp{#2}})}%
         }%
   \fi
   \ifAC@starred\else\AC@logged{#2}\fi}
```

\acp
\@acp
\Acp
\@Acp

The first time an acronym is accessed Full Names (FNs) is printed. The next time just (FNs).The optional variable is being passed to `\acfp`, in case it is used.

```
\newcommand*{\acp}{\AC@starredfalse\protect\@acp}%
\WithSuffix\newcommand\acp*{\AC@starredtrue\protect\@acp}%

\newcommand*{\Acp}{\AC@starredfalse\protect\@Acp}%
\WithSuffix\newcommand\Acp*{\AC@starredtrue\protect\@Acp}%

\newcommand{\@acp}[2][\AC@linebreakpenalty]{%
  \ifAC@dua
     \ifAC@starred\aclp*{#2}\else\aclp{#2}\fi%
  \else
   \expandafter\ifx\csname AC@\AC@prefix#2\endcsname\AC@used
      \ifAC@starred\acsp*{#2}\else\acsp{#2}\fi%
   \else
      \ifAC@starred\acfp*[#1]{#2}\else\acfp[#1]{#2}\fi%
   \fi
  \fi}

\newcommand{\@Acp}[2][\AC@linebreakpenalty]{%
  \ifAC@dua
     \ifAC@starred\Aclp*{#2}\else\Aclp{#2}\fi%
  \else
   \expandafter\ifx\csname AC@\AC@prefix#2\endcsname\AC@used
      \ifAC@starred\acsp*{#2}\else\acsp{#2}\fi%
   \else
```

```
      \ifAC@starred\Acfp*[#1]{#2}\else\Acfp[#1]{#2}\fi%
   \fi
  \fi}
```

`\acfi` `\acfia` The Full Name is printed in italics and the abbreviated is printed in upshape. The optional variable determines the penalty for a line break.

`\Acfi` `\Acfia`

```
\newcommand*{\acfi}{\AC@starredfalse\protect\acfia}%
\WithSuffix\newcommand\acfi*{\AC@starredtrue\protect\acfia}%

\newcommand*{\Acfi}{\AC@starredfalse\protect\Acfia}%
\WithSuffix\newcommand\Acfi*{\AC@starredtrue\protect\Acfia}%

\newcommand{\acfia}[2][\AC@linebreakpenalty]{%
  \texorpdfstring{\protect\@acfi[#1]{#2}}{{\AC@acl{#2}} (#2)}}

\newcommand{\Acfia}[2][\AC@linebreakpenalty]{%
   \texorpdfstring{\protect\@Acfi[#1]{#2}}{{\AC@Acl{#2}} (#2)}}

\newcommand*{\@acfi}[2][\AC@linebreakpenalty]{%
     \acffont{%
         \AC@placelabel{#2}{\itshape\AC@acl{#2}}%
         \nolinebreak[#1] %
         \acfsfont{(\acsfont{\AC@acs{#2}})}%
     }%
     \ifAC@starred\else\AC@logged{#2}\fi}

\newcommand*{\@Acfi}[2][\AC@linebreakpenalty]{%
     \acffont{%
         \AC@placelabel{#2}{\itshape\AC@Acl{#2}}%
         \nolinebreak[#1] %
         \acfsfont{(\acsfont{\AC@acs{#2}})}%
     }%
     \ifAC@starred\else\AC@logged{#2}\fi}
```

`\acused` Marks the acronym as used. Don't confuse this with `\acronymused`!

```
\newcommand{\acused}[1]{%
\global\expandafter\let\csname AC@\AC@prefix#1\endcsname\AC@used%
\AC@addtoAC@clearlist{#1}}
```

`\acsu` `\acsua` Print the short form of the acronym and mark it as used.

```
\newcommand*{\acsu}{\AC@starredfalse\protect\acsua}%
\WithSuffix\newcommand\acsu*{\AC@starredtrue\protect\acsua}%

\newcommand{\acsua}[1]{%
   \ifAC@starred\acs*{#1}\else\acs{#1}\fi\acused{#1}}
```

`\aclu` `\aclua` `\Aclu` `\Aclua` Print the long form of the acronym and mark it as used.

```
\newcommand*{\aclu}{\AC@starredfalse\protect\aclua}%
\WithSuffix\newcommand\aclu*{\AC@starredtrue\protect\aclua}%

\newcommand*{\Aclu}{\AC@starredfalse\protect\Aclua}%
\WithSuffix\newcommand\Aclu*{\AC@starredtrue\protect\Aclua}%
```

```
\newcommand{\aclua}[1]{%
   \ifAC@starred\acl*{#1}\else\acl{#1}\fi\acused{#1}}
\newcommand{\Aclua}[1]{%
   \ifAC@starred\Acl*{#1}\else\Acl{#1}\fi\acused{#1}}
\endinput
```
⟨/acronym⟩

That's it.