1 Basics

The suffix package has the purpose of making it easy to define and maintain command variants like \macro* and even \macro\/ or similar. It requires e\TeX version 2 for its work. The suffixes are fetched with \futurelet, so things like \bgroup and \{ can’t be distinguished. In addition, the efficiency depends on the complexity of the suffix’ definition, so you should preferably use characters or short commands as a suffix. A suffixed command itself counts as short for this purpose.

How does a suffix definition look like?

The general form is

\WithSuffix\WithSuffix\langle prefixed definition \rangle \langle macro \rangle \langle suffix \rangle ...

where \langle prefixed definition \rangle is something like \xdef, \global\let or similar. Recognised prefixes are \global, \long, \protected (the latter is rarely useful, as the original definition already is robust), and \expandafter (with its ‘natural’ meaning), specially recognized commands are \gdef and \xdef. Other commands can be used as long as they are suitable as an undelimited macro argument. This means they must either be a single token like \newcommand or brace-enclosed like \{\newcommand*. \langle macro \rangle can be a macro or an active character. It should be a single token suitable for assignment with \let. \langle suffix \rangle can be something like a single letter such as \* or \[. 

For example, assume that a command \snarf already exists and we want to define a variant \snarf[\langle optarg \rangle]. Then we can do this with

\WithSuffix\long\def\snarf[#1]{\langle Definition using \#1 \rangle}

That’s pretty much it. Oh, only when a command is recognised as having a prefix \global or being \xdef or \gdef will the corresponding redefinitions be done globally. This is rarely a concern.
and if you need to refer to the original unsuffixed macro, you can access it as
\csname NoSuffixName\endcsname

2 The driver file for the documentation

Installation is done by bigfoot.ins, so look there for more information for that. Here comes the documentation driver.
1 \documentclass{ltxdoc}
2 \usepackage{hyperref}
3 \usepackage{suffix}
4 \begin{document}
5 \OnlyDescription
6 \DocInput{suffix.dtx}
7 \end{document}
8 \end

3 Implementation

First we announce the package and check for eTeX 2.
1 \ProvidesPackage{suffix}[2006/07/15 1.5a Variant command support]
2 \ifcase\ifx\eTeXversion\@undefined \@ne\fi
3 \ifnum\eTeXversion<\tw@ \@ne\fi\z@
4 \else
5 \PackageError{suffix}{This package requires eTeX version 2}%
6 \if\You might try to use the 'elatex' command.\fi
7 \fi

Then we define the WithSuffix command. We use \temptokena to collect prefixes and let WSF@global to \global for global definitions.
19 \def\WithSuffix{\@temptokena{}}\let\WSF@global\relax
20 \WSF@sfx

After checking all prefixes and stuff (we'll fill in this missing link later), we add the defining command itself to the token list, place (macro) into \reserved@a and fetch \suffix into \reserved@b.
21 \long\def\WSF@sfx#1#2{\WSF@append{#1}\def\reserved@{#2}%%
22 \afterassignment\WSF@decsuff \WSF@gobblenext}
23 \def\WSF@append{\@temptokena\expandafter{\the\@temptokena#1}}
25 \def\WSF@gobblenext{\let\reserved@b= }
While we are at it, let us define the macro names to use for suffixed and unsuffixed \emph{macro}.

27 \long\def\SuffixName#1{WSF:\string#1 \meaning}
28 \def \NoSuffixName{WSF:\string}

\WSF@decsuff We first check whether the macro has already been suffixed. If it hasn’t, we save a copy of it and redefine it in terms of \WSF@suffixcheck.

29 \def \WSF@decsuff{\ifcsname \expandafter\NoSuffixName\reserved@a\endcsname
30 \else \WSF@global\expandafter\let\csname \expandafter\NoSuffixName\reserved@a
31 \expandafter\endcsname \reserved@a
32 \long\def\reserved@c##1{\WSF@global\protected\def ##1{\WSF@suffixcheck##1}}%
33 \expandafter\reserved@c\reserved@a
34 \fi}

Once we have done that, we are ready for calling the definition command on the suffixed \emph{macro}.

39 \WSF@global
40 \the\expandafter\@temptokena\csname \expandafter \SuffixName\reserved@a\reserved@b\endcsname}

\WSF@suffixcheck We now do the runtime code. This is executed in a group of its own in order not to interfere with any other macros.

43 \def \WSF@suffixcheck#1{\begingroup\def \reserved@a{#1}\
44 \futurelet \reserved@b \WSF@suffixcheckii}

\WSF@suffixcheckii After assigning the \emph{suffix} to \reserved@b, we split into the case of known and unknown suffix. We don’t code this inline, since \reserved@ in a false conditional might confuse \TeX if it happens to be something like \texttt{\iffalse} itself.

45 \def \WSF@suffixcheckii{\iffalse \expandafter\@temptokena\csname \expandafter \SuffixName\reserved@a\reserved@b\endcsname
46 \else \WSF@global
47 \the\expandafter\@temptokena\csname \expandafter \SuffixName\reserved@a\reserved@b\endcsname
48 \afterassignment\endgroup
49 \expandafter\aftergroup
50 \csname \expandafter \SuffixName\reserved@a\reserved@b\endcsname
51 \WSF@gobblenext}

\WSF@suffixcheckiii Calling the macros is reasonably straightforward, we just have to take care not to close the group at the wrong time.

53 \def \WSF@suffixcheckiii{%
54 \afterassignment\endgroup
55 \expandafter\aftergroup
56 \csname \expandafter \SuffixName\reserved@a\reserved@b\endcsname
57 \WSF@gobblenext}
Now we just augment \SF@sfx to recognize all prefixes and global commands:

\WSF@sfx
\WithSuffix\def\WSF@sfx\long{\WSF@append\long\WSF@sfx}
\WithSuffix\def\WSF@sfx\global{\let\WSF@global\global\WSF@sfx}
\WithSuffix\def\WSF@sfx\protected{\WSF@append\protected\WSF@sfx}
\WithSuffix\edef\WSF@sfx\gdef{\let\WSF@global\global\noexpand\csname\NoSuffixName\WSF@sfx\endcsname\def}
\WithSuffix\edef\WSF@sfx\xdef{\let\WSF@global\global\noexpand\csname\NoSuffixName\WSF@sfx\endcsname\edef}