

The `morewrites` package: Always room for a new `\write`

Bruno Le Floch

2024/01/05

Contents

1	<code>morewrites</code> documentation	1
1.1	Commands defined or altered by <code>morewrites</code>	2
1.2	Known deficiencies and open questions	3
2	<code>morewrites</code> implementation	3
2.1	Overview of relevant T _E X facts	3
2.2	Preliminaries	5
2.2.1	Copying some commands	5
2.2.2	Variants	5
2.2.3	Variables	6
2.2.4	Verbosity	8
2.2.5	Helpers for auxiliary file	8
2.2.6	Parsing and other helpers	10
2.3	Writing	11
2.3.1	Redefining <code>\immediate</code>	12
2.3.2	Immediate actions	12
2.3.3	Delayed actions	16
2.3.4	Shipout business	18
2.3.5	Hook at the very end	21
2.4	Redefining commands	21
2.4.1	Modified <code>\newwrite</code>	21
2.5	User commands and keys	22
	Index	24

1 morewrites documentation

This L^AT_EX package is a solution for the error “no room for a new `\write`”, which occurs when a document reserves too many streams to write data to various auxiliary files. It is in principle possible to rewrite other packages so that they are less greedy on resources, but that is often unpractical for the end-user. Instead, `morewrites` hooks at the lowest level (T_EX primitives).

Simply add the line `\usepackage{morewrites}` near the beginning of your L^AT_EX file’s preamble: the “no room for a new `\write`” error should vanish. If it does not, please contact me so that I can correct the problem. This can be done by posting a question on the tex.stackexchange.com question and answers website, logging an issue on GitHub (<https://github.com/blefloch/latex-morewrites>), or emailing me a minimal file showing the problem.

Notes.

- This package loads the `expl3` package, hence the `l3kernel` bundle needs to be up to date.
- This package uses an auxiliary file, `⟨job name⟩.mw`, which can safely be deleted. The package only overwrites this auxiliary file if it is empty, and otherwise uses a modified file name, obtained by adding an integer to the name (previously it was obtained by appending additional copies of `.mw` to the name). Such files can be safely deleted. Be careful though to not delete a Maple worksheet by accident when cleaning up your files.
- LuaT_EX allows 128 `\write` streams, so this package does nothing (with a warning) when used with LuaT_EX.

1.1 Commands defined or altered by `morewrites`

<hr/> <code>\morewritessetup</code> <hr/>	<code>\morewritessetup {⟨key-value list⟩}</code>
<hr/> New: 2014-07-26 <hr/>	Sets the options described by the <code>⟨key-value list⟩</code> .
<hr/> <code>allocate</code> <hr/>	<code>\morewritessetup { allocate = ⟨integer⟩ }</code>
<hr/> New: 2017-04-10 <hr/>	Sets to (at least) <code>⟨integer⟩</code> the number of <code>\write</code> streams allocated to the inner workings of <code>morewrites</code> . By default this is zero but increasing this value to 10 (or so) may speed up <code>morewrites</code> .
<hr/> <code>file</code> <hr/>	<code>\morewritessetup { file = ⟨file name⟩ }</code>
<hr/> New: 2014-07-26 Updated: 2024-01-05 <hr/>	Sets (globally) the name of the file which will be used by internal processes of <code>morewrites</code> . The file name is <code>\jobname.mw</code> by default (technically, <code>\c_sys_jobname_str.mw</code>). Contrarily to earlier versions of <code>morewrites</code> non-empty files will not be overwritten; this design choice may lead to unwanted <code>.mw</code> files remaining.
<hr/> <code>verbose</code> <hr/>	<code>\morewritessetup { verbose }</code>
<hr/> New: 2024-01-05 <hr/>	This boolean option (<code>false</code> by default) makes the package write to the terminal all of the operations that it performs. This can render <code>morewrites</code> useful for debugging some file-writing operations.

<hr/> <code>\newwrite</code> <hr/>	This macro is redefined by <code>morewrites</code> . Since <code>morewrites</code> allows more than 16 write streams, it removes the corresponding restrictions in <code>\newwrite</code> .
Updated: 2015-08-01	
<hr/>	
	TeXhackers note: The revised <code>\newwrite</code> allocate stream numbers starting at 129. This might break some code that expects stream numbers to be less than 16.
<hr/> <code>\immediate</code> <hr/>	This primitive is altered by <code>morewrites</code> , to detect a following <code>\write</code> or <code>\openout</code> or <code>\closeout</code> and perform the appropriate action.
Updated: 2015-08-01	
<hr/>	
<code>\openout</code> <code>\write</code> <code>\closeout</code> <hr/>	These three primitives are altered by <code>morewrites</code> so that they accept stream numbers outside the normal range [0, 15] and open/write/close files as appropriate.
Updated: 2017-04-20	TeXhackers note: System calls using <code>\write18</code> are detected and forwarded to the engine.
<hr/>	
<code>\shipout</code> <hr/>	This primitive is altered by <code>morewrites</code> to ensure that delayed <code>\openout</code> , <code>\write</code> and <code>\closeout</code> commands are performed at <code>\shipout</code> time, and in the correct order.

1.2 Known deficiencies and open questions

See the bug tracker <https://github.com/blefloch/latex-morewrites/issues/> for a list of issues with `morewrites`.

The package code is not good expl3 code. *Do not take this package as an example of how to code with expl3; go and see Joseph Wright's `siunitx` instead.* It uses `\...:D` primitives directly (the `:D` stands for “do not use”). This is unavoidable in order to hook into the primitives `\immediate`, `\write`, *etc.* and to keep a very strong control on what every command does.

2 morewrites implementation

```
<*package>
1 \RequirePackage {primargs} [2024/01/05]
2 \ProvidesExplPackage
3   {morewrites} {2024/01/05} {} {Always room for a new write}
   Quit early under LuaTeX.
4 \sys_if_engine luatex:T
5   {
6     \cs_new_protected:Npn \morewritessetup #1 { }
7     \msg_new:nnn { morewrites } { luatex }
8     { The~morewrites~package~is~unnecessary~in~LuaTeX. }
9     \msg_warning:nn { morewrites } { luatex }
10    \tex_endinput:D
11  }%
12 <@@=morewrites>
```

2.1 Overview of relevant T_EX facts

The aim of the `morewrites` package is to lift T_EX’s restriction of only having 16 files open for writing at the same time. This requires patching the primitives `\immediate`, `\openout`, `\write`, `\closeout`, and `\shipout`, and the macro `\newwrite` present in plain T_EX and L^AT_EX 2_ε.

Note that doing the same for `\read` streams is impossible due to the `\ifeof` primitive: that primitive cannot be replaced by a macro without breaking nesting of conditionals.

The `morewrites` package should be loaded as early as possible, so that any package loaded later uses the redefined macros instead of the primitives. However, the format (plain T_EX or L^AT_EX 2_ε) and the `expl3` programming language are always loaded before `morewrites`, and their interaction must be carefully monitored.

Henceforth, “T_EX stream” will refer to stream numbers in the range $[0, 15]$ provided to T_EX’s write primitives, while “user stream” will denote stream numbers in $[0, 15] \cup [129, \infty)$ manipulated by the redefined `\openout`, `\write`, `\closeout`, and `\newwrite`. A user stream in $[0, 15]$ (reserved by L^AT_EX 2_ε or allocated by `expl3`) is mapped to the same T_EX stream number, while a user stream in $[129, \infty)$ is mapped to a T_EX stream according to the property list (with integer keys and values) `\l__morewrites_write_prop`. Stream numbers 16, 17 and 18 are unused because `\write16` is often used to write to the terminal, and `\write18` sends its argument to a shell. The stream number 128 is also often used like 16 to avoid distinguishing LuaT_EX. Rather than special-casing it we skip directly to larger stream numbers.

The primitives `\openout`, `\write`, and `\closeout` expect to be followed by an $\langle integer \rangle$, normally in the range $[0, 15]$, then some further arguments.

```
\openout  $\langle integer \rangle$   $\langle equals \rangle$   $\langle file name \rangle$ 
\write  $\langle integer \rangle$   $\langle filler \rangle$   $\langle general text \rangle$ 
\closeout  $\langle integer \rangle$ 
```

All of the primitives above perform full expansion of all tokens when looking for their operands.

- $\langle integer \rangle$ denotes an integer in any form that T_EX accepts as the right-hand side of a primitive integer assignment of the form `\count0= $\langle integer \rangle$` ;
- $\langle equals \rangle$ is an arbitrary (optional) number of explicit or implicit space characters, an optional explicit equal sign of category other, and further (optional) explicit or implicit space characters;
- $\langle file name \rangle$ is an arbitrary sequence of explicit or implicit characters with arbitrary category codes (except active characters, which are expanded before reaching T_EX’s mouth), ending either with a space character (character code 32, arbitrary non-active category code, explicit or implicit), which is removed, or with a non-expandable token, with some care needed for the case of a `\notexpanded:` expandable token;
- $\langle filler \rangle$ is an arbitrary combination of tokens whose meaning is `\relax` or whose category code is 10;
- $\langle general text \rangle$ is formed of braced tokens, starting with an explicit or implicit begin-group character, and ending with the matching explicit end-group character (both

with any character code), with an equal number of explicit begin-group and end-group characters in between: this is precisely the right-hand side of an assignment of the form `\toks0=\langle general text \rangle`.

The `morewrites` package redefines these three control sequences to expect a user stream number rather than a \TeX stream number as the $\langle integer \rangle$, then map such a user stream to a \TeX stream to call the primitive with the appropriate argument. The primitive `\immediate` must also be redefined to detect `\openout`, `\write`, and `\closeout` and make them immediate, while still working with other primitives that can be made immediate. Finally, `\newwrite` must be patched to allocate stream numbers beyond 15.

A few comments on the behaviour of primitives concerning the $\langle integer \rangle$ (\TeX stream). The `\openout` primitive trigger errors if the $\langle integer \rangle$ is not in $[0, 15]$. The primitive `\write` outputs to the log if the $\langle integer \rangle$ is negative, and to the terminal if the \TeX stream is closed or greater than 15, with the exception of `\write18` which runs code in a shell. The `\closeout` primitive triggers an error if the $\langle integer \rangle$ is not in $[0, 15]$ and silently do nothing if the \TeX stream is not open, with the exception of `\closeout18` which causes a segfault at least in some versions.

By default, `\openout`, `\write` and `\closeout` are recorded in a whatsit node in the current list, and will be performed when the box containing the whatsit node is sent to the final pdf, *i.e.*, at “shipout” time. In particular, the $\langle general text \rangle$ for the `\write` primitive is expanded at shipout time. This behaviour may be modified by putting `\immediate` before any of these three primitives to force \TeX to perform the action immediately instead of recording it in a whatsit node.

Since the `\openout`, `\write`, and `\closeout` primitives operate at `\shipout` time, we will have to hook into this primitive too. It expects to be followed by a box specification, for instance `\box\langle integer \rangle` or `\hbox{\langle material to typeset \rangle}`.

Finally, the `\newwrite` macro expects one token as its argument, and defines this token (with `\chardef`) to be an integer corresponding to the first available (\TeX) write stream. This must be extended to allocate higher (user) streams.

2.2 Preliminaries

2.2.1 Copying some commands

```

\__morewrites_tex_immediate:w
\__morewrites_tex_openout:w
\__morewrites_tex_write:w
\__morewrites_tex_closeout:w
13 \cs_new_eq:NN \__morewrites_tex_immediate:w \tex_immediate:D
14 \cs_new_eq:NN \__morewrites_tex_openout:w \tex_openout:D
15 \cs_new_eq:NN \__morewrites_tex_write:w \tex_write:D
16 \cs_new_eq:NN \__morewrites_tex_closeout:w \tex_closeout:D

(End of definition for \__morewrites_tex_immediate:w and others.)

\__morewrites_tex_newwrite:N Copy \newwrite but making sure that it is not \outer. This copy will not be affected
by redefinitions of \newwrite later on.
17 \exp_args:NNf \cs_new_protected:Npn \__morewrites_tex_newwrite:N
18 { \exp_args:NNc \exp_after:wN \exp_stop_f: { newwrite } }

(End of definition for \__morewrites_tex_newwrite:N.)

```

2.2.2 Variants

We need these variants.

```

19 \cs_generate_variant:Nn \prop_gpop:NnNT { NV }
20 \cs_generate_variant:Nn \prop_gput:Nnn { NVx }
21 \cs_generate_variant:Nn \tl_gput_right:Nn { Nv }

```

2.2.3 Variables

<code>\l_morewrites_internal_tl</code>	Used for temporary scratch purposes.
<code>\l_morewrites_internal_seq</code>	<pre> 22 \tl_new:N \l_morewrites_internal_tl 23 \seq_new:N \l_morewrites_internal_seq </pre> <p>(End of definition for <code>\l_morewrites_internal_tl</code> and <code>\l_morewrites_internal_seq</code>.)</p>
<code>_morewrites_tmp:w</code>	Used for temporary definitions.
	<pre> 24 \cs_new_eq:NN _morewrites_tmp:w ? </pre> <p>(End of definition for <code>_morewrites_tmp:w</code>.)</p>
<code>\l_morewrites_verbose_bool</code>	When this boolean is set true, <code>morewrites</code> will print to the terminal all of the operations that it does.
	<pre> 25 \bool_new:N \l_morewrites_verbose_bool </pre> <p>(End of definition for <code>\l_morewrites_verbose_bool</code>.)</p>
<code>\g_morewrites_later_int</code>	The integer <code>\g_morewrites_later_int</code> labels the various non-immediate operations in the order in which they appear in the source. We can never reuse a number because there is no way to know if a whatsit was recorded in a box register, which could be reused in a shipped-out box:
	<pre> \ vbox_set:Nn \l_my_box { \ iow_shipout_x:Nn \c_term_iow {<text>} } \shipout \copy \l_my_box \shipout \copy \l_my_box </pre> <p>will print <code><text></code> to the terminal twice.</p> <pre> 26 \int_new:N \g_morewrites_later_int </pre> <p>(End of definition for <code>\g_morewrites_later_int</code>.)</p>
<code>\g_morewrites_write_seq</code>	Keep track of T _E X stream numbers managed by <code>morewrites</code> that are currently not in use as user streams.
	<pre> 27 \seq_new:N \g_morewrites_write_seq </pre> <p>(End of definition for <code>\g_morewrites_write_seq</code>.)</p>
<code>\g_morewrites_write_prop</code>	Map user streams to T _E X streams.
	<pre> 28 \prop_new:N \g_morewrites_write_prop </pre> <p>(End of definition for <code>\g_morewrites_write_prop</code>.)</p>
<code>\g_morewrites_write_file_prop</code>	Map user streams with no associated T _E X streams to file names.
	<pre> 29 \prop_new:N \g_morewrites_write_file_prop </pre> <p>(End of definition for <code>\g_morewrites_write_file_prop</code>.)</p>

`\l__morewrites_code_tl` Stores the code to run after finding a user stream, in `__morewrites_get_user:n`.

```
30 \tl_new:N \l__morewrites_code_tl
```

(End of definition for \l__morewrites_code_tl.)

`\l__morewrites_user_int` The user stream number following redefined primitives is stored in `\l__morewrites_user_int` (see `__morewrites_get_user:N`). The corresponding TeX stream number is eventually stored in `\l__morewrites_tstr_tl` (a token list).

```
31 \int_new:N \l__morewrites_user_int
```

```
32 \tl_new:N \l__morewrites_tstr_tl
```

(End of definition for \l__morewrites_user_int and \l__morewrites_tstr_tl.)

`\l__morewrites_tstr_token` This token is given as an argument to `__morewrites_tex_newwrite:N`.

```
33 \cs_new_eq:NN \l__morewrites_tstr_token ?
```

(End of definition for \l__morewrites_tstr_token.)

`\s__morewrites` A recognizable version of `\scan_stop:`. This is inspired by¹ scan marks (see the `l3quark` module of L^AT_EX3), but `\scan_new:N` is not used directly, since it has been made available in L^AT_EX3 too recently.

```
34 \cs_new_eq:NN \s__morewrites \scan_stop:
```

(End of definition for \s__morewrites.)

`\g__morewrites_iow` The expansion that `\write` performs is impossible to emulate (in X_ƎTeX at least) with anything else than `\write`. We will write on the stream `\g__morewrites_iow` to the file `\g__morewrites_tmp_file_tl` and read back from it in the stream `\g__morewrites_ior` for things to work properly. Unfortunately, this means that the file is repeatedly opened and closed, leaving a trace of that in the log.

`\g__morewrites_ior`

```
35 \newwrite \g__morewrites_iow
```

```
36 \newread \g__morewrites_ior
```

(End of definition for \g__morewrites_iow and \g__morewrites_ior.)

`\g__morewrites_tmp_file_tl` Temporary file used to do the correct expansion for each `\write`. Boolean indicating whether we have already checked that the file can be used by `morewrites`: before using a file, the `morewrites` package now checks it is empty, so as to avoid clobbering user data.

`\g__morewrites_tmp_file_bool`

```
37 \tl_new:N \g__morewrites_tmp_file_tl
```

```
38 \bool_new:N \g__morewrites_tmp_file_bool
```

```
39 \bool_gset_false:N \g__morewrites_tmp_file_bool
```

(End of definition for \g__morewrites_tmp_file_tl and \g__morewrites_tmp_file_bool.)

`\g__morewrites_group_level_int` The group level when `\shipout` is called: this is used to distinguish between explicit boxes and box registers.

```
40 \int_new:N \g__morewrites_group_level_int
```

(End of definition for \g__morewrites_group_level_int.)

`\g__morewrites_shipout_box` The page to be shipped out.

```
41 \box_new:N \g__morewrites_shipout_box
```

(End of definition for \g__morewrites_shipout_box.)

¹Historically, this might have happened the other way around, since the author of this package is also on the L^AT_EX3 Team.

2.2.4 Verbosity

`__morewrites_verbose:n` Messages to put in the terminal if the `verbose` option is selected.

```

42 \cs_new_protected:Npn \__morewrites_verbose:n #1
43   { \bool_if:NT \l__morewrites_verbose_bool { \iow_term:e { morewrites:~#1 } } }

(End of definition for \__morewrites_verbose:n.)

```

2.2.5 Helpers for auxiliary file

`__morewrites_set_file:n` Sets `\g__morewrites_tmp_file_tl` to the given value (initially `\c_sys_jobname_str.mw`). We do not yet expand, delaying that to the time where we start opening/closing the file, in case `#1` contains something that has not yet been fixed. Mark that the file has not been checked.

```

44 \cs_new_protected:Npn \__morewrites_set_file:n #1
45   {
46     \bool_gset_false:N \g__morewrites_tmp_file_bool
47     \tl_gset:Nn \g__morewrites_tmp_file_tl {#1}
48   }

(End of definition for \__morewrites_set_file:n.)

```

`__morewrites_empty_file:n` Empties a file by opening it and closing it right away. This is used when performing `\immediate \openout`. It is also used to ensure the file used by `morewrites` is left empty. We do this every time the auxiliary file is used, in case that run ends with an error mid-document.

```

49 \cs_new_protected:Npn \__morewrites_empty_file:n #1
50   {
51     \__morewrites_tex_immediate:w \__morewrites_tex_openout:w
52     \g__morewrites_iow = {#1} \scan_stop:
53     \__morewrites_tex_immediate:w \__morewrites_tex_closeout:w
54     \g__morewrites_iow
55   }

(End of definition for \__morewrites_empty_file:n.)

```

`__morewrites_if_file_trivial:nTF` True if the file does not exist, or if it is empty. Only the TF variant is defined. We set `__morewrites_tmp:w` to `\prg_return_true:` or `\prg_return_false:` within the group and use it after cleaning up. The first eof test is true if the file does not exist. Then we read one line, the second eof test is true if the file was empty (it is false if the file contained anything, even a single space).

```

56 \prg_new_conditional:Npnn \__morewrites_if_file_trivial:n #1 { TF }
57   {
58     \group_begin:
59     \tex_openin:D \g__morewrites_ior = {#1}
60     \if_eof:w \g__morewrites_ior
61       \cs_gset_eq:NN \__morewrites_tmp:w \prg_return_true:
62     \else:
63       \int_set:Nn \tex_endlinechar:D { -1 }
64       \tex_readline:D \g__morewrites_ior to \l__morewrites_internal_tl
65       \if_eof:w \g__morewrites_ior
66         \cs_gset_eq:NN \__morewrites_tmp:w \prg_return_true:
67       \else:
68         \cs_gset_eq:NN \__morewrites_tmp:w \prg_return_false:

```



```

69     \fi:
70     \fi:
71     \tex_closein:D \g__morewrites_ior
72     \group_end:
73     \__morewrites_tmp:w
74 }

```

(End of definition for __morewrites_if_file_trivial:nTF.)

__morewrites_chk_file: Expand the file name in \g__morewrites_tmp_file_tl once and for all. Check that the file does not exist or is blank. If not, try another file name obtained as follows: if it ends with .mw pick up any number that lies just before .mw and increment it, and otherwise just add .mw at the end of the file. This avoids clobbering files that the user would not want to lose.

```

75 \cs_new_protected:Npn \__morewrites_chk_file:
76 {
77     \tl_gset:Ne \g__morewrites_tmp_file_tl
78     { \tl_to_str:e { \g__morewrites_tmp_file_tl } }
79     \__morewrites_if_file_trivial:nTF { \g__morewrites_tmp_file_tl }
80     { \bool_gset_true:N \g__morewrites_tmp_file_bool }
81     {
82         \__morewrites_chk_file_aux:
83         \msg_warning:nnxx { morewrites } { file-exists }
84         { \g__morewrites_tmp_file_tl } { \l__morewrites_internal_tl }
85         \tl_gset_eq:NN \g__morewrites_tmp_file_tl \l__morewrites_internal_tl
86         \__morewrites_chk_file:
87     }
88 }
89 \cs_new_protected:Npn \__morewrites_chk_file_aux:
90 {
91     \regex_extract_once:nVNTF
92     { \A (\D*) (\d*) .mw \Z } \g__morewrites_tmp_file_tl \l__morewrites_internal_seq
93     {
94         \tl_set:Ne \l__morewrites_internal_tl
95         {
96             \seq_item:Nn \l__morewrites_internal_seq { 2 }
97             \int_eval:n { \seq_item:Nn \l__morewrites_internal_seq { 3 } + 1 }
98             .mw
99         }
100     }
101     { \tl_set:Ne \l__morewrites_internal_tl { \g__morewrites_tmp_file_tl .mw } }
102 }
103 \msg_new:nnnn { morewrites } { file-exists }
104 { File~'#1'~exists,~using~'#2'~instead. }
105 {
106     The~file~'#1'~exists~and~was~not~created~by~this~version~of~the~
107     'morewrites'~package.~Please~move~or~delete~that~file,~or~provide~
108     another~file~name~by~adding
109     \\ \\
110     \iow_indent:n { \iow_char:N\morewritessetup-{~file~~other-name~} }
111     \\ \\
112     to~your~source~file.~In~the~meantime,~the~file~'#2'~will~be~used.
113 }

```

(End of definition for __morewrites_chk_file:.)

2.2.6 Parsing and other helpers

`__morewrites_equals_file:N` Most of the parsing for primitive arguments is done using `primargs`, except for one case we care about: after its $\langle number \rangle$ argument, the `\openout` primitive expects an $\langle equals \rangle$ (optional spaces and =) and a $\langle file name \rangle$.

```

114 \cs_new_protected:Npn \__morewrites_equals_file:N #1
115 {
116   \group_begin:
117   \tex_aftergroup:D \primargs_get_input_file_name:N
118   \tex_aftergroup:D #1
119   \primargs_remove_equals:N \group_end:
120 }
```

(End of definition for `__morewrites_equals_file:N`.)

`__morewrites_get_user:n` `primargs` commands only take N-type arguments, but we often need to find an integer, save it in `\l__morewrites_user_int`, and run some code `#1`. This is analogous to `\primargs_get_number:N`.

```

121 \cs_new_protected:Npn \__morewrites_get_user:n #1
122 {
123   \tl_set:Nn \l__morewrites_code_tl {#1}
124   \tex_afterassignment:D \l__morewrites_code_tl
125   \l__morewrites_user_int =
126 }
```

(End of definition for `__morewrites_get_user:n`.)

`__morewrites_user_to_tstr:NTF` The goal is to go from a user stream `\l__morewrites_user_int` to a TeX stream `\l__morewrites_tstr_tl` (it defaults to the user stream). Streams less than 129 are not managed by `morewrites`: actual TeX streams in $[0, 15]$; negative for writing to log; 16, 17, 128 for writing to terminal; 18 for shell escape. Larger stream numbers are looked up in the property list `#1`, namely `\g__morewrites_write_prop`. If present, use the corresponding value as the TeX stream, otherwise run the `false` branch.

```

127 \cs_new_protected:Npn \__morewrites_user_to_tstr:NTF #1
128 {
129   \tl_set:NV \l__morewrites_tstr_tl \l__morewrites_user_int
130   \int_compare:nNnTF { \l__morewrites_user_int } < { 129 }
131   { \use_i:nn }
132   { \prop_get:NVNTF #1 \l__morewrites_user_int \l__morewrites_tstr_tl }
133 }
```

(End of definition for `__morewrites_user_to_tstr:NTF`.)

`\l__morewrites_collect_next_int` When encountering very large `\write` statements we may need to collect many lines. This can easily become an $O(n^2)$ task, and here we make sure that it remains around $O(n \log n)$, with a large constant unfortunately. Each of the token lists `\l__morewrites_k_tl` is empty or contains 2^k lines. As lines accumulate, they move to token lists with larger values of k , and eventually all are combined. The integer `\l__morewrites_collect_next_int` is (one plus) the maximal k among non-empty token lists.

```

134 \int_new:N \l__morewrites_collect_next_int
135 \cs_new_protected:Npn \__morewrites_collect:x #1
136 {
```

```

137 \tl_set:Nx \l__morewrites_internal_tl {#1}
138 \__morewrites_collect_aux:cf { l__morewrites_0_tl } { 1 }
139 }
140 \cs_new_protected:Npn \__morewrites_collect_aux:Nn #1#2
141 {
142   \int_compare:nNnT {#2} > \l__morewrites_collect_next_int
143   {
144     \tl_clear_new:N #1
145     \int_set:Nn \l__morewrites_collect_next_int {#2}
146   }
147   \tl_if_empty:NTF #1
148   { \tl_set_eq:NN #1 \l__morewrites_internal_tl }
149   {
150     \tl_put_left:No \l__morewrites_internal_tl {#1}
151     \tl_clear:N #1
152     \__morewrites_collect_aux:cf { l__morewrites_#2_tl }
153     { \int_eval:n { #2 + 1 } }
154   }
155 }
156 \cs_generate_variant:Nn \__morewrites_collect_aux:Nn { cf }
157 \cs_new_protected:Npn \__morewrites_collect_gput_right:N #1
158 {
159   \int_compare:nNnF \l__morewrites_collect_next_int = 0
160   {
161     \int_decr:N \l__morewrites_collect_next_int
162     \tl_gput_right:Nv #1
163     {
164       l__morewrites_
165       \int_use:N \l__morewrites_collect_next_int
166       _tl
167     }
168     \__morewrites_collect_gput_right:N #1
169   }
170 }
171 \cs_generate_variant:Nn \__morewrites_collect_gput_right:N { c }

```

(End of definition for \l__morewrites_collect_next_int and others.)

__morewrites_user_tl_name:n The name of a global token list variable holding the text of a given user stream.

```

172 \cs_new:Npn \__morewrites_user_tl_name:n #1
173 { g__morewrites_iow_ \int_eval:n {#1} _tl }

```

(End of definition for __morewrites_user_tl_name:n.)

2.3 Writing

We can hold on to material while a file is being written and only write it in one go once the file closes, to avoid using a stream throughout.

At any given time, each user stream may point to an open T_EX stream, given in \g__morewrites_write_prop, or may point to a token list that will eventually be written to a file whose file name is stored in \g__morewrites_write_file_prop, or may be closed.

When a user stream points to a token list rather than a T_EX stream, any material to be written must be written to our temporary file and read back in to apply the same expansion as \write does.

Another difficulty is that users may mix immediate and non-immediate operations. The biggest difficulty comes from the possibility of copying boxes containing delayed actions. If we ever produced a whatsit `\write⟨number⟩{⟨text⟩}` then the T_EX stream `⟨number⟩` would have to be reserved forever, as as copies of the box containing this delayed actions may be shipped out at any later point in the document.

Each delayed action is thus saved in a separate numbered token list and `\write\g__morewrites_iow{⟨number⟩}` is inserted instead of the delayed action. At each `\shipout`, the stream `\g__morewrites_iow` is opened, to catch the `⟨number⟩` of each action that should be performed at this `\shipout`.

2.3.1 Redefining `\immediate`

To accomodate the `\immediate` primitive, our versions of `\openout`, `\write` and `\closeout` will take the form

```
\s__morewrites \use_i:nn {⟨code for delayed action⟩}
{⟨code for immediate action⟩}
⟨further code⟩
```

The leading `\s__morewrites` allows the redefined `\immediate` to detect these redefined primitives, and to run the `⟨code for immediate action⟩` instead of the `⟨code for delayed action⟩` which is run by default. In both cases, any `⟨further code⟩` is run.

T_EX's `\immediate` primitive raises a flag which is cancelled after T_EX sees a non-expandable token. We use `\primargs_read_x_token:N` to find the next non-expandable token then test for `\openout`, `\write`, and `\closeout`. More precisely we test for the marker `\s__morewrites` and run the appropriate code as described above. Otherwise we call the primitive, for cases where the next token is `\pdfobj` or similar. In contrived situations involving nonsensical uses of `\noexpand` after `\immediate`, this code does not perfectly match how T_EX expands.

```
174 \cs_new_protected:Npn \__morewrites_immediate:w
175   { \primargs_read_x_token:N \__morewrites_immediate_auxii: }
176 \cs_new_protected:Npn \__morewrites_immediate_auxii:
177   {
178     \token_if_eq_meaning:NNTF \g_primargs_token \s__morewrites
179     { \__morewrites_immediate_auxiii:N }
180     {
181       \__morewrites_verbose:n
182       { \tl_to_str:n { \immediate } \token_to_meaning:N \g_primargs_token }
183       \__morewrites_tex_immediate:w
184     }
185   }
186 \cs_new_protected:Npn \__morewrites_immediate_auxiii:N #1
187   { \str_if_eq:nnTF { #1 } { \s__morewrites } { \use_iii:nnn } { #1 } }
```

(End of definition for `__morewrites_immediate:w`, `__morewrites_immediate_auxii:`, and `__morewrites_immediate_auxiii:N`.)

2.3.2 Immediate actions

The `\openout`, `\write`, and `\closeout` primitive can be either delayed or immediate. In all cases they begin by looking for a user stream. In this subsection we implement the immediate versions only.

```

\__morewrites_closeout:w
\__morewrites_closeout_now:
\__morewrites_closeout_now_silent:
\__morewrites_closeout_now:nn

```

In the immediate case `__morewrites_closeout_now:`, there are three cases. The stream may point to a T_EX stream, in which case it is closed, removed from `\g__morewrites_write_prop`, and put back in the list of usable streams. The stream may point to a token list, in which case that token list should be written to the appropriate file. The stream may be closed, in which case nothing happens. The auxiliary `__morewrites_closeout_now:nn` writes the material collected so far for a given user stream #1 to the file #2. This uses the T_EX stream `\g__morewrites_iow`. The token list consists of multiple `\immediate \write \g__morewrites_iow {<text>}` statements because that is the only safe way to obtain new lines. We do not remove the stream/file pair from `\g__morewrites_write_file_prop`.

```

188 \cs_new_protected:Npn \__morewrites_closeout:w
189 {
190   \s__morewrites
191   \use_i:nn
192   { \__morewrites_get_user:n { \__morewrites_closeout_later: } }
193   { \__morewrites_get_user:n { \__morewrites_closeout_now: } }
194 }
195 \cs_new_protected:Npn \__morewrites_closeout_now:
196 {
197   \__morewrites_verbose:n { \tl_to_str:n { \immediate \closeout } \int_use:N \l__morewrites
198   \__morewrites_closeout_now_silent:
199 }
200 \cs_new_protected:Npn \__morewrites_closeout_now_silent:
201 {
202   \__morewrites_user_to_tstr:NTF \g__morewrites_write_prop
203   {
204     \__morewrites_tex_immediate:w \__morewrites_tex_closeout:w \l__morewrites_tstr_tl \ex
205     \int_compare:nNnF { \l__morewrites_tstr_tl } = { \l__morewrites_user_int }
206     {
207       \prop_gremove:NV \g__morewrites_write_prop \l__morewrites_user_int
208       \seq_gput_left:NV \g__morewrites_write_seq \l__morewrites_tstr_tl
209     }
210   }
211   {
212     \prop_gpop:NVNT \g__morewrites_write_file_prop \l__morewrites_user_int \l__morewrites
213     { \__morewrites_closeout_now:nn { \l__morewrites_user_int } { \l__morewrites_intern
214   }
215 }
216 \cs_new_protected:Npn \__morewrites_closeout_now:nn #1#2
217 {
218   \__morewrites_tex_immediate:w \__morewrites_tex_openout:w \g__morewrites_iow = {#2}
219   \group_begin:
220     \int_set:Nn \tex_newlinechar:D { -1 }
221     \tl_use:c { \__morewrites_user_tl_name:n {#1} }
222     \tl_gclear:c { \__morewrites_user_tl_name:n {#1} }
223   \group_end:
224   \__morewrites_tex_immediate:w \__morewrites_tex_closeout:w \g__morewrites_iow
225 }

```

(End of definition for `__morewrites_closeout:w` and others.)

```

\__morewrites_openout:w
\__morewrites_openout_now:n
\__morewrites_openout_now_silent:n

```

In the immediate case find a file name, then allocate a T_EX stream if possible, and otherwise point the user stream to a token list. In all cases, close the stream to avoid

losing any material that T_EX would have written, and empty the file by opening and closing it (actually that's done automatically by the primitive).

```

226 \cs_new_protected:Npn \__morewrites_openout:w
227 {
228   \s__morewrites
229   \use_i:nn
230   { \__morewrites_get_user:n { \__morewrites_openout_later:w } }
231   { \__morewrites_get_user:n { \__morewrites_equals_file:N \__morewrites_openout_now:n } }
232 }
233 \cs_new_protected:Npn \__morewrites_openout_now:n #1
234 {
235   \__morewrites_verbose:n
236   {
237     \tl_to_str:n { \immediate\openout }
238     \int_use:N \l__morewrites_user_int
239     \c_space_tl = ~ {#1}
240   }
241   \__morewrites_openout_now_silent:n {#1}
242 }
243 \cs_new_protected:Npn \__morewrites_openout_now_silent:n #1
244 {
245   \__morewrites_closeout_now_silent:
246   \int_compare:nNnTF { \l__morewrites_user_int } < { 129 }
247   {
248     \__morewrites_tex_immediate:w \__morewrites_tex_openout:w \l__morewrites_user_int
249     = { \tl_to_str:n {#1} }
250   }
251   {
252     \seq_gpop:NNTF \g__morewrites_write_seq \l__morewrites_tstr_tl
253     {
254       \prop_gput:NVV \g__morewrites_write_prop \l__morewrites_user_int \l__morewrites_tstr_tl
255       \__morewrites_tex_immediate:w \__morewrites_tex_openout:w \l__morewrites_tstr_tl
256       = { \tl_to_str:n {#1} }
257     }
258     {
259       \__morewrites_empty_file:n {#1}
260       \prop_gput:NVx \g__morewrites_write_file_prop \l__morewrites_user_int
261       { \tl_to_str:n {#1} }
262       \tl_gc_clear_new:c { \__morewrites_user_tl_name:n { \l__morewrites_user_int } }
263     }
264   }
265 }

```

(End of definition for __morewrites_openout:w, __morewrites_openout_now:n, and __morewrites_openout_now_silent:n.)

__morewrites_write:w
 __morewrites_write_now:w
 __morewrites_write_now:n

In the immediate case we use __morewrites_write_now_open:n if the stream points to a token list, and otherwise use the primitive, with the dummy stream 16 if closed (the text is then written to the terminal).

```

266 \cs_new_protected:Npn \__morewrites_write:w
267 {
268   \s__morewrites
269   \use_i:nn
270   { \__morewrites_get_user:n { \__morewrites_write_later:w } }

```

```

271     { \_morewrites_get_user:n { \_morewrites_write_now:w } }
272   }
273 \cs_new_protected:Npn \_morewrites_write_now:w
274 {
275   \_morewrites_user_to_tstr:NTF \g\_morewrites_write_prop
276   {
277     \int_compare:nNnT \l\_morewrites_user_int = { 18 } { \use_iii:nnn }
278     \int_compare:nT { -1 < \l\_morewrites_user_int < 16 }
279     {
280       \_morewrites_verbose:n
281       {
282         \tl_to_str:n { \immediate \write }
283         \int_use:N \l\_morewrites_user_int
284       }
285     }
286     \_morewrites_tex_immediate:w \_morewrites_tex_write:w \l\_morewrites_tstr_tl \exp_s
287   }
288   { \primargs_get_general_text:N \_morewrites_write_now:n }
289 }
290 \cs_new_protected:Npn \_morewrites_write_now:n #1
291 {
292   \prop_get:NVNTF \g\_morewrites_write_file_prop \l\_morewrites_user_int \l\_morewrites_int
293   {
294     \_morewrites_verbose:n
295     {
296       \tl_to_str:n { \immediate \write }
297       \int_use:N \l\_morewrites_user_int
298       \tl_to_str:n { ~ {#1} }
299     }
300     \_morewrites_write_now_open:n {#1}
301   }
302   {
303     \_morewrites_verbose:n
304     {
305       \tl_to_str:n { \immediate \write }
306       \int_use:N \l\_morewrites_user_int
307       \tl_to_str:n { ~ (closed) ~ {#1} }
308     }
309     \_morewrites_tex_immediate:w \_morewrites_tex_write:w 16 {#1}
310   }
311 }

```

(End of definition for _morewrites_write:w, _morewrites_write_now:w, and _morewrites_write_now:n.)

_morewrites_write_now_open:n
_morewrites_write_now_loop:

Only \write itself can emulate how \write expands tokens, because # don't have to be doubled, and because the \newlinechar has to be changed to new lines. Hence, we start by writing #1 to a file (after making sure we are allowed to alter it), yielding some lines. The lines are then read one at a time using ϵ -TeX's \readline with \endlinechar set to -1 to avoid spurious characters. Each line becomes a \immediate \write statement added to a token list whose name is constructed using _morewrites_user_tl_name:n. This token list will be called when it is time to actually write to the file. At that time, \newlinechar will be -1, so that writing each line will produce no extra line.

```

312 \cs_new_protected:Npn \_morewrites_write_now_open:n #1

```

```

313 {
314   \bool_if:NF \g__morewrites_tmp_file_bool { \__morewrites_chk_file: }
315   \__morewrites_tex_immediate:w \__morewrites_tex_openout:w
316   \g__morewrites_iow = { \g__morewrites_tmp_file_tl }
317   \__morewrites_tex_immediate:w \__morewrites_tex_write:w
318   \g__morewrites_iow {#1}
319   \__morewrites_tex_immediate:w \__morewrites_tex_closeout:w
320   \g__morewrites_iow
321   \group_begin:
322     \int_set:Nn \tex_endlinechar:D { -1 }
323     \tex_openin:D \g__morewrites_ior = { \g__morewrites_tmp_file_tl }
324     \__morewrites_write_now_loop:
325     \tex_closein:D \g__morewrites_ior
326     \__morewrites_collect_gput_right:c
327     { \__morewrites_user_tl_name:n { \l__morewrites_user_int } }
328   \group_end:
329   \__morewrites_empty_file:n { \g__morewrites_tmp_file_tl }
330 }
331 \cs_new_protected:Npn \__morewrites_write_now_loop:
332 {
333   \tex_readline:D \g__morewrites_ior to \l__morewrites_internal_tl
334   \ior_if_eof:NF \g__morewrites_ior
335   {
336     \__morewrites_collect:x
337     {
338       \__morewrites_tex_immediate:w \__morewrites_tex_write:w
339       \g__morewrites_iow { \l__morewrites_internal_tl }
340     }
341     \__morewrites_write_now_loop:
342   }
343 }

```

(End of definition for __morewrites_write_now_open:n and __morewrites_write_now_loop:.)

2.3.3 Delayed actions

__morewrites_later:n Store the action to be done at shipout in a token list, and non-immediately write the label \g__morewrites_later_int of the output operation to the temporary file.

```

344 \cs_new_protected:Npn \__morewrites_later:n #1
345 {
346   \int_gincr:N \g__morewrites_later_int
347   \tl_const:cx
348   {
349     c__morewrites_later_
350     \int_use:N \g__morewrites_later_int
351     _tl
352   }
353   {
354     \int_set:Nn \exp_not:N \l__morewrites_user_int
355     { \exp_not:N \l__morewrites_user_int }
356     \exp_not:n {#1}
357   }
358   \exp_args:NNx \__morewrites_tex_write:w \g__morewrites_iow
359   { '( \int_use:N \g__morewrites_later_int ) }

```



```

360 }
361 \cs_new_protected:Npn \__morewrites_later_do:n #1
362 { \tl_use:c { c__morewrites_later_ \int_eval:n {#1} _tl } }

```

(End of definition for __morewrites_later:n and __morewrites_later_do:n.)

__morewrites_closeout_later: If the user stream is a T_EX stream, use the primitive, otherwise save __morewrites_closeout_now_silent: for later.

```

363 \cs_new_protected:Npn \__morewrites_closeout_later:
364 {
365   \__morewrites_verbose:n { \tl_to_str:n { \closeout (later) ~ } \int_use:N \l__morewrites_
366   \int_compare:nNnTF \l__morewrites_user_int < { 129 }
367     { \__morewrites_tex_closeout:w \l__morewrites_user_int }
368     { \__morewrites_later:n { \__morewrites_closeout_now_silent: } }
369 }

```

(End of definition for __morewrites_closeout_later:.)

__morewrites_openout_later:w If the user stream is a T_EX stream use the primitive, otherwise find a file name and call __morewrites_openout_now_silent:n later.

```

\__morewrites_openout_later:n
370 \cs_new_protected:Npn \__morewrites_openout_later:w
371 {
372   \int_compare:nNnTF \l__morewrites_user_int < { 129 }
373   {
374     \__morewrites_verbose:n { \tl_to_str:n { \openout (later) ~ } \int_use:N \l__morewrites_
375     \__morewrites_tex_openout:w \l__morewrites_user_int
376   }
377   { \__morewrites_equals_file:N \__morewrites_openout_later:n }
378 }
379 \cs_new_protected:Npn \__morewrites_openout_later:n #1
380 {
381   \__morewrites_verbose:n
382   {
383     \tl_to_str:n { \openout (later)~ }
384     \int_use:N \l__morewrites_user_int
385     \c_space_tl = ~ {#1}
386   }
387   \__morewrites_later:n { \__morewrites_openout_now_silent:n {#1} }
388 }

```

(End of definition for __morewrites_openout_later:w and __morewrites_openout_later:n.)

__morewrites_write_later:w For T_EX streams use the primitive, otherwise find a general text and save it for later; the auxiliary is very similar to __morewrites_write_now:w.

```

\__morewrites_write_later:n
\__morewrites_write_later_aux:n
389 \cs_new_protected:Npn \__morewrites_write_later:w
390 {
391   \int_compare:nNnTF \l__morewrites_user_int < { 129 }
392   {
393     \__morewrites_verbose:n { \tl_to_str:n { \write (later)~ } \int_use:N \l__morewrites_
394     \__morewrites_tex_write:w \l__morewrites_user_int
395   }
396   { \primargs_get_general_text:N \__morewrites_write_later:n }
397 }
398 \cs_new_protected:Npn \__morewrites_write_later:n #1

```

```

399 {
400   \__morewrites_verbose:n
401   {
402     \tl_to_str:n { \write (later)~ }
403     \int_use:N \l__morewrites_user_int
404     \tl_to_str:n { ~ {#1} }
405   }
406   \__morewrites_later:n { \__morewrites_write_later_aux:n {#1} }
407 }
408 \cs_new_protected:Npn \__morewrites_write_later_aux:n
409 {
410   \__morewrites_user_to_tstr:NTF \g__morewrites_write_prop
411   { \__morewrites_tex_immediate:w \__morewrites_tex_write:w \l__morewrites_tstr_tl \exp_s
412   {
413     \prop_get:NVNTF \g__morewrites_write_file_prop \l__morewrites_user_int \l__morewrites
414     { \__morewrites_write_now_open:n }
415     { \__morewrites_tex_immediate:w \__morewrites_tex_write:w 16 \exp_stop_f: }
416   }
417 }

```

(End of definition for __morewrites_write_later:w, __morewrites_write_later:n, and __morewrites_write_later_aux:n.)

2.3.4 Shipout business

In this section, we hook into the \shipout primitive, and redefine it to first build a box with the material to ship out, then perform

```

\__morewrites_before_shipout:
<primitive shipout> <collected box>
\__morewrites_after_shipout:

```

Each delayed output operation has been replaced by \write \g__morewrites_iow {‘(<operation number>)’}. The delimiters we chose to put around numbers must be at least two distinct characters on the left (then \tex_newlinechar:D cannot be equal to the delimiter), and at least one non-digit character on the right.

__morewrites_before_shipout: Immediately before the shipout, we must open the writing stream \g__morewrites_iow (after making sure we are allowed to alter the auxiliary file).

```

418 \cs_new_protected:Npn \__morewrites_before_shipout:
419 {
420   \bool_if:NF \g__morewrites_tmp_file_bool { \__morewrites_chk_file: }
421   \__morewrites_tex_immediate:w \__morewrites_tex_openout:w
422   \g__morewrites_iow = { \g__morewrites_tmp_file_tl }
423 }

```

(End of definition for __morewrites_before_shipout:.)

__morewrites_after_shipout: Immediately after all the \writes are performed, close the file, then read the file with \endlinechar set to \newlinechar² to get exactly the original characters that have been written, possibly with extra characters between ‘(…)’ groups. The file is then read

²Note that the \newlinechar used by \writes at \shipout time are those in effect when the page is shipped out, *i.e.*, just after the closing brace of the \shipout construction, which is exactly where we have added this hook.

with all the appropriate category codes set up (no other character can appear in the file). The looping auxiliary `__morewrites_after_shipout_loop:ww` extracts the *operation* numbers from the file, and makes a token list out of those. This token list is then used in a mapping function to perform the appropriate `\write` operations. Note that those operations may reuse the file, so we have to fully parse the file before moving on.

```

424 \cs_new_protected:Npn \__morewrites_after_shipout:
425 {
426   \__morewrites_tex_immediate:w \__morewrites_tex_closeout:w
427   \g__morewrites_iow
428   \group_begin:
429     \int_set_eq:NN \tex_endlinechar:D \tex_newlinechar:D
430     \char_set_catcode_other:n { \tex_endlinechar:D }
431     \tl_map_inline:nn { '(0123456789) }
432       { \char_set_catcode_other:n {'##1} }
433     \tex_everyeof:D { '() \exp_not:N }
434     \tl_set:Nx \l__morewrites_internal_tl
435       {
436         \exp_after:wN \__morewrites_after_shipout_loop:ww
437         \tex_input:D { \g__morewrites_tmp_file_tl }
438       }
439     \__morewrites_empty_file:n { \g__morewrites_tmp_file_tl }
440     \exp_args:NNo
441     \group_end:
442     \tl_map_function:nN { \l__morewrites_internal_tl } \__morewrites_later_do:n
443   }
444 \cs_new:Npn \__morewrites_after_shipout_loop:ww #1 '( #2 )
445 {
446   \tl_if_empty:nF {#2}
447   {
448     {#2}
449     \__morewrites_after_shipout_loop:ww
450   }
451 }

```

(End of definition for `__morewrites_after_shipout:` and `__morewrites_after_shipout_loop:ww`.)

`__morewrites_shipout:w`
`__morewrites_shipout_i:`
`__morewrites_shipout_ii:`

Grab the shipped out box using `\setbox` and regain control using `\afterassignment`. There are two cases: either the box is given as `\box` or `\copy` followed by a number, in which case `__morewrites_shipout_i:` is inserted afterwards at the same group level, or the box is given as `\hbox` (or `\vtop` and so on) and an additional `\aftergroup` is needed to reach a point where we can use the box saved in `\g__morewrites_shipout_box`.

```

452 \cs_new_protected:Npn \__morewrites_shipout:w
453 {
454   \int_gset_eq:NN \g__morewrites_group_level_int \tex_currentgrouplevel:D
455   \tex_afterassignment:D \__morewrites_shipout_i:
456   \tex_global:D \tex_setbox:D \g__morewrites_shipout_box
457 }
458 \cs_new_protected:Npn \__morewrites_shipout_i:
459 {
460   \int_compare:nNnTF { \g__morewrites_group_level_int }
461     = { \tex_currentgrouplevel:D }
462     { \__morewrites_shipout_ii: }
463     { \tex_aftergroup:D \__morewrites_shipout_ii: }

```

```

464 }
465 \cs_new_protected:Npn \__morewrites_shipout_ii:
466 {
467   \__morewrites_before_shipout:
468   \__morewrites_tex_shipout:w \tex_box:D \g__morewrites_shipout_box
469   \__morewrites_after_shipout:
470 }

(End of definition for \__morewrites_shipout:w, \__morewrites_shipout_i:, and \__morewrites_
shipout_ii:.)

```

\shipout

The task is now to locate the shipout primitive, which may have been renamed and hooked into by many different packages loaded before `morewrites`. Any of those control sequences which are equal to the primitive are redefined to do `__morewrites_shipout:w` instead. If the primitive is not located at all, the fallback is to hook into the control sequence `\shipout`.

```

471 \cs_gset_protected:Npn \__morewrites_tmp:w #1
472 {
473   \cs_if_exist:NF \__morewrites_tex_shipout:w
474     { \cs_new_eq:NN \__morewrites_tex_shipout:w #1 }
475   \cs_gset_eq:NN #1 \__morewrites_shipout:w
476 }
477 \tl_map_inline:nn
478 {
479   \xyrealshipout@
480   \org@shipout
481   \PDFSYNCship@ut@ld
482   \CROP@shipout
483   \@soORI
484   \tex_shipout:D
485   \zwpl@Hship
486   \o@shipout@TP
487   \LL@shipout
488   \Shipout
489   \GXTorg@shipout
490   \AtBegShi@OrgShipout
491   \AtBeginShipoutOriginalShipout
492   \minidocument@orig@shipout
493   \shipout
494 }
495 {
496   \str_if_eq:eeT
497     { \cs_meaning:N #1 }
498     { \token_to_str:N \shipout }
499     { \__morewrites_tmp:w #1 }
500 }
501 \cs_if_exist:NF \__morewrites_tex_shipout:w
502 {
503   \cs_new_eq:NN \__morewrites_tex_shipout:w \shipout
504   \cs_gset_eq:NN \shipout \__morewrites_shipout:w
505 }

```

(End of definition for `\shipout` and `__morewrites_tex_shipout:w`. This function is documented on page 3.)

2.3.5 Hook at the very end

`__morewrites_close_all:` At the end of the document, close all the files.

```

506 \cs_new_protected:Npn \__morewrites_close_all:
507 {
508   \prop_map_inline:Nn \g__morewrites_write_prop
509   {
510     \__morewrites_verbose:n { \tl_to_str:n { \immediate \closeout } ##1 ~ (at-end) }
511     \__morewrites_tex_immediate:w \__morewrites_tex_closeout:w ##2 \scan_stop:
512   }
513   \prop_gclear:N \g__morewrites_write_prop
514   \prop_map_function:NN \g__morewrites_write_file_prop
515   \__morewrites_closeout_now:nn
516   \prop_gclear:N \g__morewrites_write_file_prop
517 }

```

(End of definition for __morewrites_close_all:.)

`__morewrites_close_all_at_end:nw` At the end of the run, we try very hard to put some material at the `\@@end`, just in case some other very late code writes to files that are not yet closed. This is tried at most 5 times, to avoid infinite loops in case two packages compete for that last place. The four `@` become two after `l3docstrip`.

```

518 \cs_new_protected:Npn \__morewrites_close_all_at_end:nw #1#2 \@@end
519 {
520   \int_compare:nNnTF {#1} > 0
521   { #2 \__morewrites_close_all_at_end:nw { #1 - 1 } }
522   { \__morewrites_close_all: #2 }
523   \@@end
524 }
525 \AtEndDocument { \__morewrites_close_all_at_end:nw { 5 } }

```

(End of definition for __morewrites_close_all_at_end:nw.)

2.4 Redefining commands

2.4.1 Modified `\newwrite`

`\g__morewrites_alloc_write_int` Counter to allocate user streams. We used to initialize it to 18 so that the first user stream allocated by `morewrites` was 19. Indeed, 18 is reserved for shell commands and packages may expect 16 or 17 to write to the terminal. This is now changed to start allocation at 129, since some packages that do not want to distinguish LuaTeX from other engines simply use 128 as a never-open stream.

```

526 \int_new:N \g__morewrites_alloc_write_int
527 \int_gset:Nn \g__morewrites_alloc_write_int { 128 }

```

(End of definition for \g__morewrites_alloc_write_int.)

`__morewrites_newwrite:N` Reimplementation of `\newwrite` but protected and using a counter `\g__morewrites_alloc_write_int` instead of what TeX/LaTeX 2_ε use.

```

528 \cs_new_protected:Npn \__morewrites_newwrite:N #1
529 {
530   \int_gincr:N \g__morewrites_alloc_write_int
531   \int_set_eq:NN \allocationnumber \g__morewrites_alloc_write_int
532   \cs_undefine:N #1

```

```

533 \int_const:Nn #1 { \allocationnumber }
534 \wlog
535 {
536   \token_to_str:N #1
537   = \token_to_str:N \write \int_use:N \allocationnumber
538 }
539 }

```

(End of definition for _morewrites_newwrite:N.)

_morewrites_allocate:n Raise to #1 the number of \write streams allocated to morewrites.

```

540 \cs_new_protected:Npn \_morewrites\_allocate:n #1
541 {
542   \prg_replicate:nn
543   {
544     \int_max:nn { 0 }
545     {
546       (#1) - \seq_count:N \g\_morewrites\_write\_seq
547       - \prop_count:N \g\_morewrites\_write\_prop
548     }
549   }
550   {
551     \_morewrites\_tex\_newwrite:N \l\_morewrites\_tstr\_token
552     \seq_gput_right:NV \g\_morewrites\_write\_seq \l\_morewrites\_tstr\_token
553   }
554 }

```

(End of definition for _morewrites_allocate:n.)

2.5 User commands and keys

\morewritessetup Set whatever keys the user passes to \morewritessetup.

```

555 \cs_new_protected:Npn \morewritessetup #1
556 { \keys_set:nn { \_morewrites } {#1} }

```

(End of definition for \morewritessetup. This function is documented on page 2.)

file Because of our use of .initial:n, this code must appear after _morewrites_set_file:n is defined.

```

557 \keys_define:nn { \_morewrites }
558 {
559   allocate .code:n = \_morewrites\_allocate:n {#1} ,
560   file .code:n = \_morewrites\_set\_file:n {#1} ,
561   file .initial:n = \c_sys_jobname_str .mw ,
562   verbose .bool_set:N = \l\_morewrites\_verbose\_bool
563 }

```

(End of definition for file. This function is documented on page 2.)

\immediate

\openout

\write

\closeout

\newwrite

```

564 \cs_gset_eq:NN \immediate \_morewrites\_immediate:w
565 \cs_gset_eq:NN \openout \_morewrites\_openout:w
566 \cs_gset_eq:NN \write \_morewrites\_write:w
567 \cs_gset_eq:NN \closeout \_morewrites\_closeout:w
568 \cs_gset_eq:NN \newwrite \_morewrites\_newwrite:N

```

(End of definition for \immediate and others. These functions are documented on page 3.)
</package>

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.

Symbols	
<code>\\</code>	109, 110, 111
A	
<code>\A</code>	92
<code>allocate</code>	2
<code>\allocationnumber</code>	531, 533, 537
<code>\AtBeginShipoutOriginalShipout</code> ...	491
<code>\AtEndDocument</code>	525
B	
bool commands:	
<code>\bool_gset_false:N</code>	39, 46
<code>\bool_gset_true:N</code>	80
<code>\bool_if:NTF</code>	43, 314, 420
<code>\bool_new:N</code>	25, 38
box commands:	
<code>\box_new:N</code>	41
C	
char commands:	
<code>\char_set_catcode_other:n</code> ..	430, 432
<code>\closeout</code>	3, 197, 365, 510, 564
cs commands:	
<code>\cs_generate_variant:Nn</code>	19, 20, 21, 156, 171
<code>\cs_gset_eq:NN</code>	61, 66, 68, 475, 504, 564, 565, 566, 567, 568
<code>\cs_gset_protected:Npn</code>	471
<code>\cs_if_exist:NTF</code>	473, 501
<code>\cs_meaning:N</code>	497
<code>\cs_new:Npn</code>	172, 444
<code>\cs_new_eq:NN</code>	13, 14, 15, 16, 24, 33, 34, 474, 503
<code>\cs_new_protected:Npn</code>	6, 17, 42, 44, 49, 75, 89, 114, 121, 127, 135, 140, 157, 174, 176, 186, 188, 195, 200, 216, 226, 233, 243, 266, 273, 290, 312, 331, 344, 361, 363, 370, 379, 389, 398, 408, 418, 424, 452, 458, 465, 506, 518, 528, 540, 555
<code>\cs_undefine:N</code>	532
D	
<code>\D</code>	92
<code>\d</code>	92
E	
else commands:	
<code>\else:</code>	62, 67
exp commands:	
<code>\exp_after:wN</code>	18, 436
<code>\exp_args:NNc</code>	18
<code>\exp_args:NNf</code>	17
<code>\exp_args:NNn</code>	440
<code>\exp_args:NNx</code>	358
<code>\exp_not:N</code>	354, 433
<code>\exp_not:n</code>	355, 356
<code>\exp_stop_f:</code> ..	18, 204, 255, 286, 411, 415
F	
fi commands:	
<code>\fi:</code>	69, 70
<code>file</code>	2, 557
G	
group commands:	
<code>\group_begin:</code> ..	58, 116, 219, 321, 428
<code>\group_end:</code>	72, 119, 223, 328, 441
I	
if commands:	
<code>\if_eof:w</code>	60, 65
<code>\immediate</code>	3, 182, 197, 237, 282, 296, 305, 510, 564
int commands:	
<code>\int_compare:nNnTF</code> ..	130, 142, 159, 205, 246, 277, 366, 372, 391, 460, 520
<code>\int_compare:nTF</code>	278
<code>\int_const:Nn</code>	533
<code>\int_decr:N</code>	161
<code>\int_eval:n</code>	97, 153, 173, 362
<code>\int_gincr:N</code>	346, 530
<code>\int_gset:Nn</code>	527
<code>\int_gset_eq:NN</code>	454
<code>\int_max:nn</code>	544
<code>\int_new:N</code>	26, 31, 40, 134, 526
<code>\int_set:Nn</code>	63, 145, 220, 322, 354
<code>\int_set_eq:NN</code>	429, 531
<code>\int_use:N</code>	165, 197, 238, 283, 297, 306, 350, 359, 365, 374, 384, 393, 403, 537
ior commands:	
<code>\ior_if_eof:NTF</code>	334
iow commands:	
<code>\iow_char:N</code>	110
<code>\iow_indent:n</code>	110
<code>\iow_shipout_x:Nn</code>	6
<code>\iow_term:n</code>	43

<p>\c_term_iow 6</p> <p style="text-align: center;">K</p> <p>keys commands:</p> <p>\keys_define:nn 557</p> <p>\keys_set:nn 556</p> <p style="text-align: center;">M</p> <p>morewrites internal commands:</p> <p>__morewrites_after_shipout: 18, 424, 424, 469</p> <p>__morewrites_after_shipout_ loop:ww 19, 424, 436, 444, 449</p> <p>\g__morewrites_alloc_write_int 21, 526, 530, 531</p> <p>__morewrites_allocate:n 540, 540, 559</p> <p>__morewrites_before_shipout: 18, 418, 418, 467</p> <p>__morewrites_chk_file: 75, 75, 86, 314, 420</p> <p>__morewrites_chk_file_aux: . 82, 89</p> <p>__morewrites_close_all: 506, 506, 522</p> <p>__morewrites_close_all_at_ end:nw 518, 518, 521, 525</p> <p>__morewrites_closeout:w 188, 188, 567</p> <p>__morewrites_closeout_later: 192, 363, 363</p> <p>__morewrites_closeout_now: 13, 188, 193, 195</p> <p>__morewrites_closeout_now:nn 13, 188, 213, 216, 515</p> <p>__morewrites_closeout_now_ silent: . 17, 188, 198, 200, 245, 368</p> <p>\l__morewrites_code_tl .. 30, 123, 124</p> <p>__morewrites_collect:n 134, 135, 336</p> <p>__morewrites_collect_aux:Nn 134, 138, 140, 152, 156</p> <p>__morewrites_collect_gput_ right:N 134, 157, 168, 171, 326</p> <p>\l__morewrites_collect_next_int 10, 134</p> <p>__morewrites_empty_file:n 49, 49, 259, 329, 439</p> <p>__morewrites_equals_file:N 114, 114, 231, 377</p> <p>__morewrites_get_user:N 7</p> <p>__morewrites_get_user:n 6, 121, 121, 192, 193, 230, 231, 270, 271</p> <p>\g__morewrites_group_level_int 40, 454, 460</p> <p>__morewrites_if_file_trivial:n . 56</p> <p>__morewrites_if_file_trivial:nTF 56, 79</p>	<p>__morewrites_immediate:w 174, 174, 564</p> <p>__morewrites_immediate_auxii: 174, 175, 176</p> <p>__morewrites_immediate_auxiii:N 174, 179, 186</p> <p>\l__morewrites_internal_seq 22, 92, 96, 97</p> <p>\l__morewrites_internal_tl .. 22, 64, 84, 85, 94, 101, 137, 148, 150, 212, 213, 292, 333, 339, 413, 434, 442</p> <p>\g__morewrites_ior 7, 35, 59, 60, 64, 65, 71, 323, 325, 333, 334</p> <p>\g__morewrites_iow 7, 12, 13, 18, 35, 52, 54, 218, 224, 316, 318, 320, 339, 358, 422, 427</p> <p>__morewrites_later:n 344, 344, 368, 387, 406</p> <p>__morewrites_later_do:n 344, 361, 442</p> <p>\g__morewrites_later_int 6, 16, 26, 346, 350, 359</p> <p>__morewrites_newwrite:N 528, 528, 568</p> <p>__morewrites_openout:w 226, 226, 565</p> <p>__morewrites_openout_later:n 370, 377, 379</p> <p>__morewrites_openout_later:w 230, 370, 370</p> <p>__morewrites_openout_now:n 226, 231, 233</p> <p>__morewrites_openout_now_ silent:n 17, 226, 241, 243, 387</p> <p>__morewrites_set_file:n 22, 44, 44, 560</p> <p>__morewrites_shipout:w 20, 452, 452, 475, 504</p> <p>\g__morewrites_shipout_box 19, 41, 456, 468</p> <p>__morewrites_shipout_i: 19, 452, 455, 458</p> <p>__morewrites_shipout_ii: 452, 462, 463, 465</p> <p>__morewrites_tex_closeout:w 13, 16, 53, 204, 224, 319, 367, 426, 511</p> <p>__morewrites_tex_immediate:w 13, 13, 51, 53, 183, 204, 218, 224, 248, 255, 286, 309, 315, 317, 319, 338, 411, 415, 421, 426, 511</p> <p>__morewrites_tex_newwrite:N 7, 17, 17, 551</p> <p>__morewrites_tex_openout:w . 13, 14, 51, 218, 248, 255, 315, 375, 421</p> <p>__morewrites_tex_shipout:w 468, 471, 473, 474, 501, 503</p>
--	---

__morewrites_tex_write:w	13, 15, 286, 309, 317, 338, 358, 394, 411, 415
__morewrites_tmp:w	8, 24, 24, 61, 66, 68, 73, 471, 499
\g__morewrites_tmp_file_bool	37, 46, 80, 314, 420
\g__morewrites_tmp_file_tl	7-9, 37, 47, 77, 78, 79, 84, 85, 92, 101, 316, 323, 329, 422, 437, 439
\l__morewrites_tstr_tl	7, 10, 31, 129, 132, 204, 205, 208, 252, 254, 255, 286, 411
\l__morewrites_tstr_token	33, 551, 552
\l__morewrites_user_int	7, 10, 31, 125, 129, 130, 132, 197, 205, 207, 212, 213, 238, 246, 248, 254, 260, 262, 277, 278, 283, 292, 297, 306, 327, 354, 355, 365, 366, 367, 372, 374, 375, 384, 391, 393, 394, 403, 413
__morewrites_user_tl_name:n	15, 172, 172, 221, 222, 262, 327
__morewrites_user_to_tstr:NTF	127, 127, 202, 275, 410
__morewrites_verbose:n	42, 42, 181, 197, 235, 280, 294, 303, 365, 374, 381, 393, 400, 510
\l__morewrites_verbose_bool	25, 43, 562
__morewrites_write:w	266, 266, 566
\g__morewrites_write_file_prop	11, 13, 29, 212, 260, 292, 413, 514, 516
__morewrites_write_later:n	389, 396, 398
__morewrites_write_later:w	270, 389, 389
__morewrites_write_later_aux:n	389, 406, 408
__morewrites_write_now:n	266, 288, 290
__morewrites_write_now:w	17, 266, 271, 273
__morewrites_write_now_loop:	312, 324, 331, 341
__morewrites_write_now_open:n	14, 300, 312, 312, 414
\g__morewrites_write_prop	10, 11, 13, 28, 202, 207, 254, 275, 410, 508, 513, 547
\l__morewrites_write_prop	4
\g__morewrites_write_seq	27, 208, 252, 546, 552
\morewritessetup	2, 22, 6, 555
msg commands:	
\msg_new:nnn	7
\msg_new:nnnn	103
\msg_warning:nn	9
\msg_warning:nnnn	83
my commands:	
\l_my_box	6
N	
\newread	36
\newwrite	2, 35, 564
O	
\openout	3, 237, 374, 383, 564
P	
prg commands:	
\prg_new_conditional:Npnn	56
\prg_replicate:nn	542
\prg_return_false:	8, 68
\prg_return_true:	8, 61, 66
primargs commands:	
\primargs_get_general_text:N	288, 396
\primargs_get_input_file_name:N	117
\primargs_get_number:N	10
\primargs_read_x_token:N	12, 175
\primargs_remove_equals:N	119
\g_primargs_token	178, 182
prop commands:	
\prop_count:N	547
\prop_gclear:N	513, 516
\prop_get:NnNTF	132, 292, 413
\prop_gpop:NnNTF	19, 212
\prop_gput:Nnn	20, 254, 260
\prop_gremove:Nn	207
\prop_map_function:NN	514
\prop_map_inline:Nn	508
\prop_new:N	28, 29
\ProvidesExplPackage	2
R	
regex commands:	
\regex_extract_once:nnNTF	91
\RequirePackage	1
S	
scan commands:	
\scan_new:N	7
\scan_stop:	7, 34, 52, 511
scan internal commands:	
\s__morewrites	12, 34, 178, 187, 190, 228, 268
seq commands:	
\seq_count:N	546
\seq_gpop:NNTF	252
\seq_gput_left:Nn	208
\seq_gput_right:Nn	552

<code>\seq_item:Nn</code>	96, 97	<code>\tex_box:D</code>	468
<code>\seq_new:N</code>	23, 27	<code>\tex_closein:D</code>	71, 325
<code>\Shipout</code>	488	<code>\tex_closeout:D</code>	16
<code>\shipout</code>	3, 471	<code>\tex_currentgrouplevel:D</code> ...	454, 461
str commands:		<code>\tex_endinput:D</code>	10
<code>\str_if_eq:nnTF</code>	187, 496	<code>\tex_endlinechar:D</code> ..	63, 322, 429, 430
sys commands:		<code>\tex_everyeof:D</code>	433
<code>\sys_if_engine luatex:TF</code>	4	<code>\tex_global:D</code>	456
<code>\c_sys_jobname_str</code>	2, 8, 561	<code>\tex_immediate:D</code>	13
T			
TeX and L ^A T _Ε X 2 _ε commands:			
<code>\@@end</code>	21, 518, 523	<code>\tex_input:D</code>	437
<code>\@soORI</code>	483	<code>\tex_newlinechar:D</code>	18, 220, 429
<code>\afterassignment</code>	19	<code>\tex_openin:D</code>	59, 323
<code>\aftergroup</code>	19	<code>\tex_openout:D</code>	14
<code>\AtBegShi@OrgShipout</code>	490	<code>\tex_readline:D</code>	64, 333
<code>\box</code>	5, 19	<code>\tex_setbox:D</code>	456
<code>\chardef</code>	5	<code>\tex_shipout:D</code>	484
<code>\closeout</code>	3–5, 12	<code>\tex_write:D</code>	15
<code>\copy</code>	6, 19	tl commands:	
<code>\count</code>	4	<code>\c_space_tl</code>	239, 385
<code>\CROP@shipout</code>	482	<code>\tl_clear:N</code>	151
<code>\endlinechar</code>	15, 18	<code>\tl_clear_new:N</code>	144
<code>\GXTorg@shipout</code>	489	<code>\tl_const:Nn</code>	347
<code>\hbox</code>	5, 19	<code>\tl_gclear:N</code>	222
<code>\ifeof</code>	4	<code>\tl_gclear_new:N</code>	262
<code>\immediate</code>	3, 5, 8, 12, 13, 15	<code>\tl_gput_right:Nn</code>	21, 162
<code>\jobname</code>	2	<code>\tl_gset:Nn</code>	47, 77
<code>\LL@shipout</code>	487	<code>\tl_gset_eq:NN</code>	85
<code>\minidocument@orig@shipout</code>	492	<code>\tl_if_empty:NTF</code>	147
<code>\newlinechar</code>	15, 18	<code>\tl_if_empty:nTF</code>	446
<code>\newwrite</code>	2–5, 21	<code>\tl_map_function:nN</code>	442
<code>\noexpand</code>	12	<code>\tl_map_inline:nn</code>	431, 477
<code>\notexpanded:</code>	4	<code>\tl_new:N</code>	22, 30, 32, 37
<code>\o@shipout@TP</code>	486	<code>\tl_put_left:Nn</code>	150
<code>\openout</code>	3–5, 8, 10, 12	<code>\tl_set:Nn</code> ..	94, 101, 123, 129, 137, 434
<code>\org@shipout</code>	480	<code>\tl_set_eq:NN</code>	148
<code>\outer</code>	5	<code>\tl_to_str:n</code> ...	78, 182, 197, 237, 249, 256, 261, 282, 296, 298, 305, 307, 365, 374, 383, 393, 402, 404, 510
<code>\pdfobj</code>	12	<code>\tl_use:N</code>	221, 362
<code>\PDFSYNCship@ut@ld</code>	481	token commands:	
<code>\read</code>	4	<code>\token_if_eq_meaning:NNTF</code>	178
<code>\readline</code>	15	<code>\token_to_meaning:N</code>	182
<code>\relax</code>	4	<code>\token_to_str:N</code>	498, 536, 537
<code>\setbox</code>	19	U	
<code>\shipout</code>	3, 5–7, 12, 18, 20	use commands:	
<code>\toks</code>	4	<code>\use_i:nn</code>	12, 131, 191, 229, 269
<code>\vtop</code>	19	<code>\use_iii:nnn</code>	187, 277
<code>\write</code> ...	1–5, 7, 10–13, 15, 18, 19, 22	V	
<code>\xyrealshipout@</code>	479	vbox commands:	
<code>\zwpl@Hship</code>	485	<code>\vbox_set:Nn</code>	6
tex commands:		verbose	2
<code>\tex_afterassignment:D</code>	124, 455		
<code>\tex_aftergroup:D</code>	117, 118, 463		

	W	Z
\wlog		534
\write .	3, 282, 296, 305, 393, 402, 537, <u>564</u>	\Z 92