

gtkterm

1.99.0

Generated by Doxygen 1.9.1

1 GTKTerm: A GTK+ Serial Port Terminal	1
1.1 Usage	1
1.1.1 Keyboard Shortcuts	1
1.1.2 Command Line Options	1
1.1.3 Notes on RS485:	2
1.1.4 Scriptability with Signals	2
1.2 Installation	2
1.3 Uninstallation	3
1.4 License	3
2 Namespace Index	5
2.1 Namespace List	5
3 Class Index	7
3.1 Class List	7
4 File Index	9
4.1 File List	9
5 Namespace Documentation	11
5.1 meson_post_install Namespace Reference	11
5.1.1 Variable Documentation	11
5.1.1.1 install_prefix	11
5.1.1.2 schemadir	11
6 Class Documentation	13
6.1 cfgList_tag Struct Reference	13
6.1.1 Member Data Documentation	13
6.1.1.1 next	13
6.1.1.2 str	14
6.2 cfgStruct Struct Reference	14
6.2.1 Member Data Documentation	14
6.2.1.1 parameterName	14
6.2.1.2 type	15
6.2.1.3 value	15
6.3 display_config_t Struct Reference	15
6.3.1 Member Data Documentation	16
6.3.1.1 background_color	16
6.3.1.2 block_cursor	16
6.3.1.3 char_queue	16
6.3.1.4 columns	17
6.3.1.5 crlfauto	17
6.3.1.6 delay	17
6.3.1.7 echo	17

6.3.1.8 font	17
6.3.1.9 foreground_color	18
6.3.1.10 rows	18
6.3.1.11 scrollbar	18
6.3.1.12 show_cursor	18
6.3.1.13 timestamp	18
6.3.1.14 visual_bell	19
6.4 GtkTermWindow Struct Reference	19
6.4.1 Member Data Documentation	20
6.4.1.1 buffer	20
6.4.1.2 fullscreen	20
6.4.1.3 height	20
6.4.1.4 infobar	20
6.4.1.5 maximized	20
6.4.1.6 menubutton	20
6.4.1.7 message	20
6.4.1.8 parent_instance	21
6.4.1.9 status	21
6.4.1.10 toolmenu	21
6.4.1.11 width	21
6.5 macro_t Struct Reference	21
6.5.1 Detailed Description	22
6.5.2 Member Data Documentation	22
6.5.2.1 action	22
6.5.2.2 closure	22
6.5.2.3 shortcut	22
6.6 port_config_t Struct Reference	23
6.6.1 Member Data Documentation	23
6.6.1.1 bits	23
6.6.1.2 char_queue	24
6.6.1.3 disable_port_lock	24
6.6.1.4 flow_control	24
6.6.1.5 parity	24
6.6.1.6 port	24
6.6.1.7 rs485_rts_time_after_transmit	24
6.6.1.8 rs485_rts_time_before_transmit	25
6.6.1.9 speed	25
6.6.1.10 stops	25
7 File Documentation	27
7.1 buffer.c File Reference	27
7.1.1 Macro Definition Documentation	28

7.1.1.1	TIMESTAMP_SIZE	28
7.1.2	Function Documentation	28
7.1.2.1	clear_buffer()	28
7.1.2.2	create_buffer()	28
7.1.2.3	delete_buffer()	28
7.1.2.4	insert_timestamp()	29
7.1.2.5	put_chars()	29
7.1.2.6	set_clear_func()	29
7.1.2.7	set_display_func()	29
7.1.2.8	unset_clear_func()	29
7.1.2.9	unset_display_func()	29
7.1.2.10	write_buffer()	30
7.1.2.11	write_buffer_with_func()	30
7.1.3	Variable Documentation	30
7.1.3.1	clear_func	30
7.1.3.2	overlapped	31
7.1.3.3	timestamp_on	31
7.1.3.4	virt_col_pos	31
7.1.3.5	write_func	31
7.2	buffer.h File Reference	31
7.2.1	Macro Definition Documentation	32
7.2.1.1	BUFFER_SIZE	32
7.2.2	Function Documentation	32
7.2.2.1	clear_buffer()	32
7.2.2.2	create_buffer()	32
7.2.2.3	delete_buffer()	33
7.2.2.4	put_chars()	33
7.2.2.5	set_clear_func()	33
7.2.2.6	set_display_func()	33
7.2.2.7	unset_clear_func()	33
7.2.2.8	unset_display_func()	33
7.2.2.9	write_buffer()	34
7.2.2.10	write_buffer_with_func()	34
7.3	cmdline.c File Reference	34
7.3.1	Function Documentation	35
7.3.1.1	display_help()	35
7.3.1.2	read_command_line()	36
7.3.2	Variable Documentation	36
7.3.2.1	config	36
7.4	cmdline.h File Reference	37
7.4.1	Function Documentation	37
7.4.1.1	read_command_line()	37

7.5 files.c File Reference	37
7.5.1 Variable Documentation	38
7.5.1.1 default_filename	38
7.6 files.h File Reference	38
7.6.1 Function Documentation	38
7.6.1.1 add_input()	39
7.6.1.2 save_raw_file()	39
7.6.1.3 send_raw_file()	39
7.6.2 Variable Documentation	39
7.6.2.1 default_filename	39
7.6.2.2 waiting_for_char	39
7.7 gtkterm.c File Reference	40
7.7.1 Typedef Documentation	40
7.7.1.1 GtkTerm	40
7.7.1.2 GtkTermClass	41
7.7.1.3 GtkTermWindowClass	41
7.7.2 Function Documentation	41
7.7.2.1 main()	41
7.7.2.2 set_window_title()	41
7.8 gtkterm_conv.c File Reference	42
7.8.1 Function Documentation	42
7.8.1.1 load_old_configuration_from_file()	43
7.8.1.2 main()	44
7.8.1.3 show_message()	44
7.8.2 Variable Documentation	45
7.8.2.1 cfg	45
7.8.2.2 port_conf	45
7.8.2.3 term_conf	45
7.9 i18n.c File Reference	46
7.9.1 Function Documentation	46
7.9.1.1 i18n_fprintf()	46
7.9.1.2 i18n_perror()	46
7.9.1.3 i18n_printf()	47
7.9.1.4 strerror_utf8()	47
7.10 i18n.h File Reference	47
7.10.1 Macro Definition Documentation	48
7.10.1.1 I18N_H	48
7.10.2 Function Documentation	48
7.10.2.1 i18n_fprintf()	48
7.10.2.2 i18n_perror()	48
7.10.2.3 i18n_printf()	49
7.10.2.4 strerror_utf8()	49

7.11 interface.c File Reference	49
7.11.1 Function Documentation	50
7.11.1.1 show_message()	50
7.11.2 Variable Documentation	50
7.11.2.1 config	50
7.11.2.2 display	51
7.11.2.3 timestamp_on	51
7.11.2.4 virt_col_pos	51
7.12 interface.h File Reference	51
7.12.1 Macro Definition Documentation	52
7.12.1.1 ASCII_VIEW	52
7.12.1.2 HEXADECIMAL_VIEW	52
7.12.1.3 MSG_ERR	52
7.12.1.4 MSG_WRN	52
7.12.2 Function Documentation	52
7.12.2.1 show_message()	52
7.12.3 Variable Documentation	53
7.12.3.1 display	53
7.12.3.2 Text	53
7.13 macros.c File Reference	53
7.13.1 Enumeration Type Documentation	54
7.13.1.1 anonymous enum	54
7.13.2 Function Documentation	55
7.13.2.1 convert_macros_to_string()	55
7.13.2.2 convert_string_to_macros()	55
7.13.2.3 get_shortcuts()	56
7.13.2.4 macro_count()	56
7.13.2.5 remove_shortcuts()	56
7.13.3 Variable Documentation	57
7.13.3.1 macros	57
7.13.3.2 nr_of_macros	57
7.14 macros.h File Reference	57
7.14.1 Function Documentation	58
7.14.1.1 add_shortcuts()	58
7.14.1.2 convert_macros_to_string()	58
7.14.1.3 convert_string_to_macros()	59
7.14.1.4 get_shortcuts()	59
7.14.1.5 macro_count()	59
7.14.1.6 remove_shortcuts()	60
7.14.2 Variable Documentation	60
7.14.2.1 macros	60
7.15 meson_post_install.py File Reference	60

7.16 old_config.c File Reference	61
7.16.1 Function Documentation	62
7.16.1.1 create_shortcuts()	62
7.16.1.2 load_old_configuration_from_file()	62
7.16.2 Variable Documentation	63
7.16.2.1 background_alpha	63
7.16.2.2 background_blue	63
7.16.2.3 background_green	64
7.16.2.4 background_red	64
7.16.2.5 bits	64
7.16.2.6 block_cursor	64
7.16.2.7 cfg	64
7.16.2.8 columns	64
7.16.2.9 crlfauto	65
7.16.2.10 echo	65
7.16.2.11 flow	65
7.16.2.12 font	65
7.16.2.13 foreground_alpha	65
7.16.2.14 foreground_blue	65
7.16.2.15 foreground_green	66
7.16.2.16 foreground_red	66
7.16.2.17 macro_list	66
7.16.2.18 nr_of_macros	66
7.16.2.19 parity	66
7.16.2.20 port	66
7.16.2.21 rows	67
7.16.2.22 rts_time_after_tx	67
7.16.2.23 rts_time_before_tx	67
7.16.2.24 scrollbar	67
7.16.2.25 speed	67
7.16.2.26 stopbits	67
7.16.2.27 timestamp	68
7.16.2.28 visual_bell	68
7.16.2.29 wait_char	68
7.16.2.30 wait_delay	68
7.17 parsecfg.c File Reference	68
7.17.1 Function Documentation	69
7.17.1.1 cfgAllocForNewSection()	69
7.17.1.2 cfgDump()	69
7.17.1.3 cfgParse()	69
7.17.1.4 cfgSectionNameToNumber()	70
7.17.1.5 cfgSectionNumberToName()	70

7.17.1.6	cfgSetFatalFunc()	70
7.17.1.7	cfgStoreValue()	70
7.17.1.8	fetchVarFromCfgFile()	71
7.18	parsecfg.h File Reference	71
7.18.1	Macro Definition Documentation	72
7.18.1.1	PARSECFG_VERSION	72
7.18.2	Typedef Documentation	72
7.18.2.1	cfgList	72
7.18.3	Enumeration Type Documentation	72
7.18.3.1	cfgErrorCode	72
7.18.3.2	cfgFileType	73
7.18.3.3	cfgKeywordValue	73
7.18.3.4	cfgQuote	73
7.18.3.5	cfgValueType	74
7.18.4	Function Documentation	74
7.18.4.1	cfgAllocForNewSection()	74
7.18.4.2	cfgDump()	74
7.18.4.3	cfgParse()	75
7.18.4.4	cfgSectionNameToNumber()	75
7.18.4.5	cfgSectionNumberToName()	75
7.18.4.6	cfgSetFatalFunc()	76
7.18.4.7	cfgStoreValue()	76
7.18.4.8	fetchVarFromCfgFile()	76
7.19	README.md File Reference	76
7.20	resource_file.c File Reference	76
7.20.1	Macro Definition Documentation	77
7.20.1.1	CONFIGURATION_FILENAME	78
7.20.2	Enumeration Type Documentation	78
7.20.2.1	anonymous enum	78
7.20.3	Function Documentation	79
7.20.3.1	check_configuration_file()	79
7.20.3.2	config_file_init()	80
7.20.3.3	copy_configuration()	80
7.20.3.4	dump_configuration_to_cli()	81
7.20.3.5	hard_default_configuration()	82
7.20.3.6	load_configuration_from_file()	83
7.20.3.7	remove_section()	84
7.20.3.8	save_configuration_to_file()	85
7.20.3.9	set_color()	85
7.20.3.10	validate_configuration()	86
7.20.4	Variable Documentation	86
7.20.4.1	config_file	86

7.20.4.2 ConfigurationItem	87
7.21 resource_file.h File Reference	87
7.21.1 Function Documentation	88
7.21.1.1 check_configuration_file()	88
7.21.1.2 config_file_init()	89
7.21.1.3 copy_configuration()	89
7.21.1.4 dump_configuration_to_cli()	90
7.21.1.5 hard_default_configuration()	91
7.21.1.6 load_configuration_from_file()	92
7.21.1.7 remove_section()	93
7.21.1.8 save_configuration_to_file()	94
7.21.1.9 set_color()	94
7.21.1.10 validate_configuration()	95
7.21.2 Variable Documentation	95
7.21.2.1 config_file	95
7.22 serial.c File Reference	96
7.22.1 Function Documentation	96
7.22.1.1 get_port_string()	96
7.22.2 Variable Documentation	97
7.22.2.1 port_conf	97
7.22.2.2 serial_port_fd	97
7.22.2.3 termios_save	97
7.23 serial.h File Reference	97
7.23.1 Macro Definition Documentation	98
7.23.1.1 DEFAULT_BITS	98
7.23.1.2 DEFAULT_FLOW	98
7.23.1.3 DEFAULT_PARITY	98
7.23.1.4 DEFAULT_PORT	99
7.23.1.5 DEFAULT_SPEED	99
7.23.1.6 DEFAULT_STOP	99
7.23.1.7 LINE_FEED	99
7.23.1.8 POLL_DELAY	99
7.23.1.9 RECEIVE_BUFFER	99
7.23.1.10 TRANSMIT_BUFFER	99
7.23.2 Function Documentation	99
7.23.2.1 get_port_string()	100
7.23.3 Variable Documentation	100
7.23.3.1 port_conf	100
7.23.3.2 serial_port_fd	100
7.24 term_config.c File Reference	101
7.24.1 Macro Definition Documentation	101
7.24.1.1 CONFIGURATION_FILENAME	101

7.24.2 Variable Documentation	101
7.24.2.1 term_conf	101
7.25 term_config.h File Reference	102
7.25.1 Macro Definition Documentation	102
7.25.1.1 DEFAULT_CHAR	102
7.25.1.2 DEFAULT_DELAY	102
7.25.1.3 DEFAULT_DELAY_RS485	103
7.25.1.4 DEFAULT_ECHO	103
7.25.1.5 DEFAULT_FONT	103
7.25.1.6 DEFAULT_SCROLLBACK	103
7.25.2 Variable Documentation	103
7.25.2.1 term_conf	103
Index	105

Chapter 1

GTKTerm: A GTK+ Serial Port Terminal

GTKTerm is a simple, graphical serial port terminal emulator for Linux and possibly other POSIX-compliant operating systems. It can be used to communicate with all kinds of devices with a serial interface, such as embedded computers, microcontrollers, modems, GPS receivers, CNC machines and more.

1.1 Usage

1.1.1 Keyboard Shortcuts

As GTKTerm is often used like a terminal emulator, the shortcut keys are assigned to `<ctrl><shift>`, rather than just `<ctrl>`. This allows the user to send keystrokes of the form `<ctrl>X` and not have GTKTerm intercept them.

Key Combination	Effect
<code><ctrl><shift>L</code>	Clear screen
<code><ctrl><shift>R</code>	Send file
<code><ctrl><shift>Q</code>	Quit
<code><ctrl><shift>S</code>	Configure port
<code><ctrl><shift>V</code>	Paste
<code><ctrl><shift>C</code>	Copy
<code><ctrl><shift>F</code>	Find
<code><ctrl><shift>K</code>	Clear Scrollback
<code><ctrl><shift>A</code>	Select All
<code><ctrl><shift>B</code>	Send Break
<code><ctrl>B</code>	Send break
F5	Open Port
F6	Close Port
F7	Toggle DTR
F8	Toggle RTS

1.1.2 Command Line Options

See `man gtkterm` or `gtkterm --help` for more information on available command line interface options.

1.1.3 Notes on RS485:

The RS485 flow control is a software user-space emulation and therefore may not work for all configurations (won't respond quickly enough). If this is the case for your setup, you will need to either use a dedicated RS232 to RS485 converter, or look for a kernel level driver. This is an inherent limitation to user space programs.

1.1.4 Scriptability with Signals

Some microcontrollers and other embedded devices are flashed using the same serial interface that is also used for outputting debug information. To facilitate rapid development on these platforms, GTKTerm supports the following UNIX signals:

Signal	Action	Usage Example
SIGUSR1	Open Port	<code>killall -USR1 gtkterm</code>
SIGUSR2	Close Port	<code>killall -USR2 gtkterm</code>

You may find it useful to send these signals in your own firmware flashing scripts.

1.2 Installation

GTKTerm has a few dependencies-

- Gtk+4.0 (version 4.6 or higher)
- vte-gtk4 (version 0.68 or higher)
- intltool (version 0.40.0 or higher)
- libgudev (version 229 or higher)

Once these dependencies are installed, most people should simply run:

```
meson build
ninja -C build
```

To install GTKTerm system-wide, run:

```
ninja -C build install
gtk-update-icon-cache
```

If you wish to install GTKTerm someplace other than the default directory, e.g. in `/usr`, use:

```
meson build -Dprefix=/usr
```

Then build and install as usual.

1.3 Uninstallation

To uninstall GTKTerm, run:

```
ninja -C build uninstall
```

If you already deleted the `build` directory, just compile and install GTKTerm again as explained in the `previous` section with the same target location prefix (`-Dprefix`) and perform the `uninstall` step afterwards.

1.4 License

Original Code by: Julien Schmitt

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

meson_post_install	11
-------------------------------------	----

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

cfgList_tag	13
cfgStruct	14
display_config_t	15
GtkTermWindow	19
macro_t	
Define macro structure type	21
port_config_t	23

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

buffer.c	27
buffer.h	31
cmdline.c	34
cmdline.h	37
files.c	37
files.h	38
gtkterm.c	40
gtkterm_conv.c	42
i18n.c	46
i18n.h	47
interface.c	49
interface.h	51
macros.c	53
macros.h	57
meson_post_install.py	60
old_config.c	61
parsecfg.c	68
parsecfg.h	71
resource_file.c	76
resource_file.h	87
serial.c	96
serial.h	97
term_config.c	101
term_config.h	102

Chapter 5

Namespace Documentation

5.1 meson_post_install Namespace Reference

Variables

- **install_prefix** = os.environ['MESON_INSTALL_PREFIX']
- **schemadir** = os.path.join(**install_prefix**, 'share', 'glib-2.0', 'schemas')

5.1.1 Variable Documentation

5.1.1.1 install_prefix

```
meson_post_install.install_prefix = os.environ['MESON_INSTALL_PREFIX']
```

5.1.1.2 schemadir

```
meson_post_install.schemadir = os.path.join( install_prefix, 'share', 'glib-2.0', 'schemas')
```

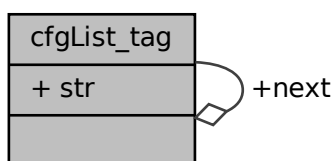

Chapter 6

Class Documentation

6.1 cfgList_tag Struct Reference

```
#include <parsecfg.h>
```

Collaboration diagram for `cfgList_tag`:



Public Attributes

- `char * str`
- `struct cfgList_tag * next`

6.1.1 Member Data Documentation

6.1.1.1 next

```
struct cfgList_tag* cfgList_tag::next
```

Referenced by `load_old_configuration_from_file()`.

6.1.1.2 str

```
char* cfgList_tag::str
```

Referenced by `load_old_configuration_from_file()`.

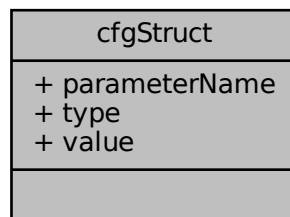
The documentation for this struct was generated from the following file:

- `parsecfg.h`

6.2 cfgStruct Struct Reference

```
#include <parsecfg.h>
```

Collaboration diagram for `cfgStruct`:



Public Attributes

- `char * parameterName`
- `cfgValueType type`
- `void * value`

6.2.1 Member Data Documentation

6.2.1.1 parameterName

```
char* cfgStruct::parameterName
```

6.2.1.2 type

```
cfgValueType cfgStruct::type
```

6.2.1.3 value

```
void* cfgStruct::value
```

The documentation for this struct was generated from the following file:

- parsecfg.h

6.3 display_config_t Struct Reference

```
#include <term_config.h>
```

Collaboration diagram for display_config_t:



Public Attributes

- gboolean **block_cursor**
- gboolean **show_cursor**
- char **char_queue**
- gboolean **echo**
- gboolean **crlfauto**
- gboolean **timestamp**
- int **delay**
- int **rows**
- int **columns**
- int **scrollback**
- gboolean **visual_bell**
- GdkRGBA **foreground_color**
- GdkRGBA **background_color**
- PangoFontDescription * **font**

6.3.1 Member Data Documentation

6.3.1.1 background_color

GdkRGBA display_config_t::background_color

Referenced by **copy_configuration()**, **dump_configuration_to_cli()**, **hard_default_configuration()**, **load_configuration_from_file()**, and **load_old_configuration_from_file()**.

6.3.1.2 block_cursor

gboolean display_config_t::block_cursor

Referenced by **dump_configuration_to_cli()**, **hard_default_configuration()**, and **load_old_configuration_from_file()**.

6.3.1.3 char_queue

char display_config_t::char_queue

Referenced by **copy_configuration()**, **dump_configuration_to_cli()**, **hard_default_configuration()**, **load_configuration_from_file()**, **load_old_configuration_from_file()**, and **read_command_line()**.

6.3.1.4 columns

```
int display_config_t::columns
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, and `load_old_configuration_from_file()`.

6.3.1.5 crlfauto

```
gboolean display_config_t::crlfauto
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, and `load_old_configuration_from_file()`.

6.3.1.6 delay

```
int display_config_t::delay
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, `read_command_line()`, and `validate_configuration()`.

6.3.1.7 echo

```
gboolean display_config_t::echo
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, and `read_command_line()`.

6.3.1.8 font

```
PangoFontDescription* display_config_t::font
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, and `validate_configuration()`.

6.3.1.9 foreground_color

GdkRGBA display_config_t::foreground_color

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, and `load_old_configuration_from_file()`.

6.3.1.10 rows

int display_config_t::rows

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, and `load_old_configuration_from_file()`.

6.3.1.11 scrollbar

int display_config_t::scrollback

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, and `load_old_configuration_from_file()`.

6.3.1.12 show_cursor

gboolean display_config_t::show_cursor

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, and `load_configuration_from_file()`.

6.3.1.13 timestamp

gboolean display_config_t::timestamp

Referenced by `dump_configuration_to_cli()`, `hard_default_configuration()`, and `load_old_configuration_from_file()`.

6.3.1.14 visual_bell

gboolean display_config_t::visual_bell

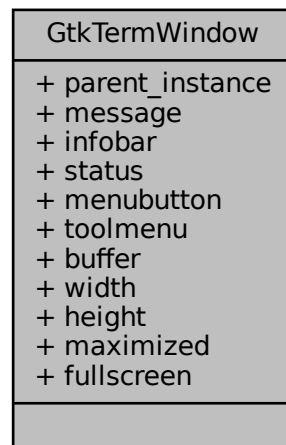
Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, and `load_old_configuration_from_file()`.

The documentation for this struct was generated from the following file:

- `term_config.h`

6.4 GtkTermWindow Struct Reference

Collaboration diagram for GtkTermWindow:



Public Attributes

- GtkApplicationWindow **parent_instance**
- GtkWidget * **message**
- GtkWidget * **infobar**
- GtkWidget * **status**
- GtkWidget * **menubutton**
- GMenuModel * **toolmenu**
- GtkTextBuffer * **buffer**
- int **width**
- int **height**
- gboolean **maximized**
- gboolean **fullscreen**

6.4.1 Member Data Documentation

6.4.1.1 buffer

`GtkTextBuffer* GtkTermWindow::buffer`

6.4.1.2 fullscreen

`gboolean GtkTermWindow::fullscreen`

6.4.1.3 height

`int GtkTermWindow::height`

6.4.1.4 infobar

`GtkWidget* GtkTermWindow::infobar`

6.4.1.5 maximized

`gboolean GtkTermWindow::maximized`

6.4.1.6 menubutton

`GtkWidget* GtkTermWindow::menubutton`

6.4.1.7 message

`GtkWidget* GtkTermWindow::message`

6.4.1.8 parent_instance

GtkApplicationWindow GtkTermWindow::parent_instance

6.4.1.9 status

GtkWidget* GtkTermWindow::status

6.4.1.10 toolmenu

GMenuModel* GtkTermWindow::toolmenu

6.4.1.11 width

int GtkTermWindow::width

The documentation for this struct was generated from the following file:

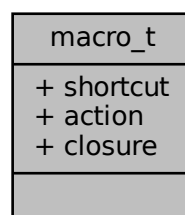
- gtkterm.c

6.5 macro_t Struct Reference

Define macro structure type.

```
#include <macros.h>
```

Collaboration diagram for macro_t:



Public Attributes

- char * **shortcut**
- char * **action**
Shortcut of the macro.
- GClosure * **closure**
Command to perform.

6.5.1 Detailed Description

Define macro structure type.

6.5.2 Member Data Documentation

6.5.2.1 action

```
char* macro_t::action
```

Shortcut of the macro.

Referenced by `convert_macros_to_string()`, `convert_string_to_macros()`, `create_shortcuts()`, and `load_↔old_configuration_from_file()`.

6.5.2.2 closure

```
GClosure* macro_t::closure
```

Command to perform.

6.5.2.3 shortcut

```
char* macro_t::shortcut
```

Referenced by `convert_macros_to_string()`, `convert_string_to_macros()`, `create_shortcuts()`, and `load_↔old_configuration_from_file()`.

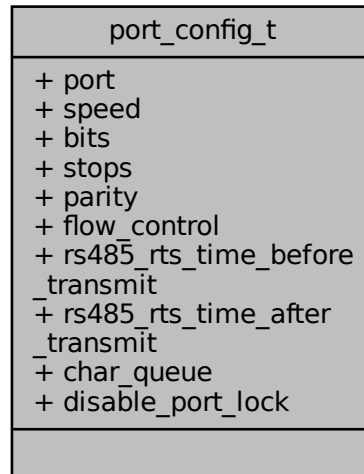
The documentation for this struct was generated from the following file:

- **macros.h**

6.6 port_config_t Struct Reference

```
#include <serial.h>
```

Collaboration diagram for port_config_t:



Public Attributes

- char **port** [256]
- long int **speed**
- int **bits**
- int **stops**
- int **parity**
- int **flow_control**
- int **rs485_rts_time_before_transmit**
- int **rs485_rts_time_after_transmit**
- char **char_queue**
- gboolean **disable_port_lock**

6.6.1 Member Data Documentation

6.6.1.1 bits

```
int port_config_t::bits
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `get_port_string()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, `read_command_line()`, and `validate_configuration()`.

6.6.1.2 char_queue

```
char port_config_t::char_queue
```

6.6.1.3 disable_port_lock

```
gboolean port_config_t::disable_port_lock
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, and `read_command_line()`.

6.6.1.4 flow_control

```
int port_config_t::flow_control
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, and `read_command_line()`.

6.6.1.5 parity

```
int port_config_t::parity
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `get_port_string()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, and `read_command_line()`.

6.6.1.6 port

```
char port_config_t::port[256]
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `get_port_string()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, and `read_command_line()`.

6.6.1.7 rs485_rts_time_after_transmit

```
int port_config_t::rs485_rts_time_after_transmit
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, and `read_command_line()`.

6.6.1.8 rs485_rts_time_before_transmit

```
int port_config_t::rs485_rts_time_before_transmit
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, and `read_command_line()`.

6.6.1.9 speed

```
long int port_config_t::speed
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `get_port_string()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, `read_command_line()`, and `validate_configuration()`.

6.6.1.10 stops

```
int port_config_t::stops
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `get_port_string()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, `read_command_line()`, and `validate_configuration()`.

The documentation for this struct was generated from the following file:

- `serial.h`

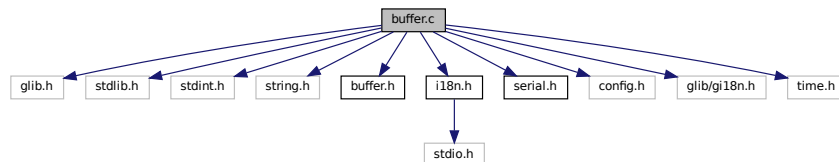
Chapter 7

File Documentation

7.1 buffer.c File Reference

```
#include <glib.h>
#include <stdlib.h>
#include <stdint.h>
#include <string.h>
#include "buffer.h"
#include "i18n.h"
#include "serial.h"
#include <config.h>
#include <glib/gi18n.h>
#include <time.h>
```

Include dependency graph for buffer.c:



Macros

- `#define` **TIMESTAMP_SIZE** 50

Functions

- void **create_buffer** (void)
- void **delete_buffer** (void)
- unsigned int **insert_timestamp** (char *buffer)
- void **put_chars** (const char *chars, unsigned int size, gboolean crlf_auto)
- void **write_buffer** (void)
- void **write_buffer_with_func** (void(*func)(const char *, unsigned int))
- void **clear_buffer** (void)
- void **set_clear_func** (void(*func)(void))
- void **unset_clear_func** (void(*func)(void))
- void **set_display_func** (void(*func)(const char *, unsigned int))
- void **unset_display_func** (void(*func)(const char *, unsigned int))

Variables

- gboolean **timestamp_on**
- char **overlapped**
- guint **virt_col_pos**
- void(* **write_func**)(const char *, unsigned int) = NULL
- void(* **clear_func**)(void) = NULL

7.1.1 Macro Definition Documentation

7.1.1.1 TIMESTAMP_SIZE

```
#define TIMESTAMP_SIZE 50
```

7.1.2 Function Documentation

7.1.2.1 clear_buffer()

```
void clear_buffer (  
    void )
```

References **clear_func**.

7.1.2.2 create_buffer()

```
void create_buffer (  
    void )
```

7.1.2.3 delete_buffer()

```
void delete_buffer (  
    void )
```


7.1.2.4 insert_timestamp()

```
unsigned int insert_timestamp (
    char * buffer )
```

7.1.2.5 put_chars()

```
void put_chars (
    const char * chars,
    unsigned int size,
    gboolean crlf_auto )
```

References **RECEIVE_BUFFER**, **timestamp_on**, and **TIMESTAMP_SIZE**.

7.1.2.6 set_clear_func()

```
void set_clear_func (
    void(*) (void) func )
```

References **clear_func**.

7.1.2.7 set_display_func()

```
void set_display_func (
    void(*) (const char *, unsigned int) func )
```

References **write_func**.

7.1.2.8 unset_clear_func()

```
void unset_clear_func (
    void(*) (void) func )
```

References **clear_func**.

7.1.2.9 unset_display_func()

```
void unset_display_func (
    void(*) (const char *, unsigned int) func )
```

References **write_func**.

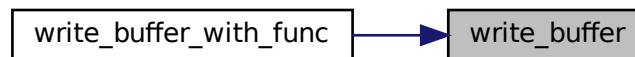
7.1.2.10 write_buffer()

```
void write_buffer (
    void )
```

References **overlapped**, and **write_func**.

Referenced by **write_buffer_with_func()**.

Here is the caller graph for this function:

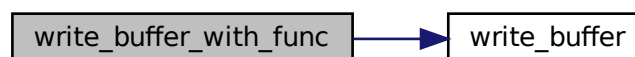


7.1.2.11 write_buffer_with_func()

```
void write_buffer_with_func (
    void(*) (const char *, unsigned int) func )
```

References **write_buffer()**, and **write_func**.

Here is the call graph for this function:



7.1.3 Variable Documentation

7.1.3.1 clear_func

```
void(* clear_func) (void) (
    void ) = NULL
```

Referenced by **clear_buffer()**, **set_clear_func()**, and **unset_clear_func()**.

7.1.3.2 overlapped

char overlapped

Referenced by **write_buffer()**.

7.1.3.3 timestamp_on

gboolean timestamp_on [extern]

Referenced by **put_chars()**.

7.1.3.4 virt_col_pos

guint virt_col_pos [extern]

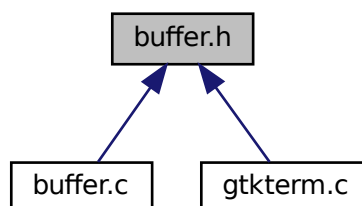
7.1.3.5 write_func

```
void(* write_func) (const char *, unsigned int) (  
    const char * ,  
    unsigned int ) = NULL
```

Referenced by **set_display_func()**, **unset_display_func()**, **write_buffer()**, and **write_buffer_with_func()**.

7.2 buffer.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- `#define BUFFER_SIZE (128 * 1024)`

Functions

- void **create_buffer** (void)
- void **delete_buffer** (void)
- void **put_chars** (const char *, unsigned int, gboolean)
- void **clear_buffer** (void)
- void **write_buffer** (void)
- void **set_display_func** (void(*func)(const char *, uint32_t))
- void **unset_display_func** (void(*func)(const char *, uint32_t))
- void **set_clear_func** (void(*func)(void))
- void **unset_clear_func** (void(*func)(void))
- void **write_buffer_with_func** (void(*func)(const char *, uint32_t))

7.2.1 Macro Definition Documentation

7.2.1.1 BUFFER_SIZE

```
#define BUFFER_SIZE (128 * 1024)
```

7.2.2 Function Documentation

7.2.2.1 clear_buffer()

```
void clear_buffer (  
    void )
```

References **clear_func**.

7.2.2.2 create_buffer()

```
void create_buffer (  
    void )
```

7.2.2.3 delete_buffer()

```
void delete_buffer (
    void )
```

7.2.2.4 put_chars()

```
void put_chars (
    const char * chars,
    unsigned int size,
    gboolean crlf_auto )
```

References **RECEIVE_BUFFER**, **timestamp_on**, and **TIMESTAMP_SIZE**.

7.2.2.5 set_clear_func()

```
void set_clear_func (
    void(*) (void) func )
```

References **clear_func**.

7.2.2.6 set_display_func()

```
void set_display_func (
    void(*) (const char *, uint32_t) func )
```

7.2.2.7 unset_clear_func()

```
void unset_clear_func (
    void(*) (void) func )
```

References **clear_func**.

7.2.2.8 unset_display_func()

```
void unset_display_func (
    void(*) (const char *, uint32_t) func )
```

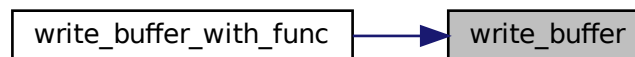
7.2.2.9 write_buffer()

```
void write_buffer (
    void )
```

References **overlapped**, and **write_func**.

Referenced by **write_buffer_with_func()**.

Here is the caller graph for this function:



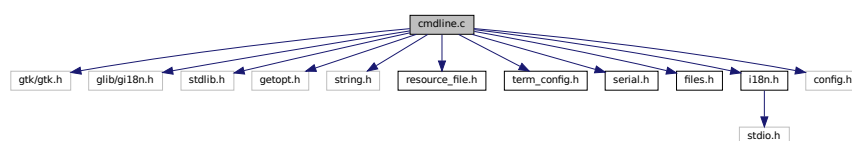
7.2.2.10 write_buffer_with_func()

```
void write_buffer_with_func (
    void(*) (const char *, uint32_t) func )
```

7.3 cmdline.c File Reference

```
#include <gtk/gtk.h>
#include <glib/gi18n.h>
#include <stdlib.h>
#include <getopt.h>
#include <string.h>
#include "resource_file.h"
#include "term_config.h"
#include "serial.h"
#include "files.h"
#include "i18n.h"
#include <config.h>
```

Include dependency graph for cmdline.c:



Functions

- void **display_help** (void)
- int **read_command_line** (int argc, char **argv, char *configuration_to_read)

Variables

- struct configuration_port **config**

7.3.1 Function Documentation

7.3.1.1 display_help()

```
void display_help (  
    void )
```

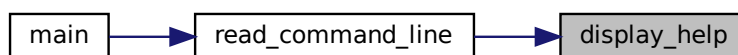
References **i18n_printf()**.

Referenced by **read_command_line()**.

Here is the call graph for this function:



Here is the caller graph for this function:



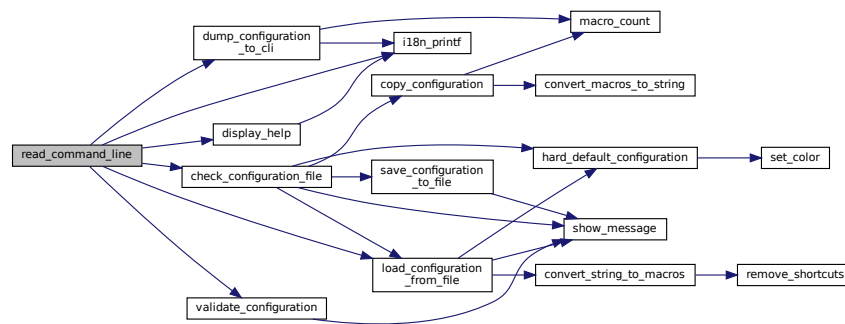
7.3.1.2 read_command_line()

```
int read_command_line (
    int argc,
    char ** argv,
    char * configuration_to_read )
```

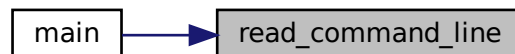
References `port_config_t::bits`, `display_config_t::char_queue`, `check_configuration_file()`, `default_↵`
`_filename`, `display_config_t::delay`, `port_config_t::disable_port_lock`, `display_help()`, `dump_↵`
`configuration_to_cli()`, `display_config_t::echo`, `port_config_t::flow_control`, `i18n_printf()`, `load_↵`
`configuration_from_file()`, `port_config_t::parity`, `port_config_t::port`, `port_conf`, `port_config_t::rs485_↵`
`_rts_time_after_transmit`, `port_config_t::rs485_rts_time_before_transmit`, `port_config_t::speed`, `port_↵`
`config_t::stops`, `term_conf`, and `validate_configuration()`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.3.2 Variable Documentation

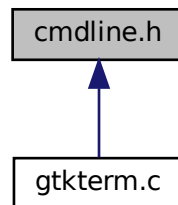
7.3.2.1 config

```
struct configuration_port config [extern]
```

Referenced by `check_configuration_file()`, and `save_configuration_to_file()`.

7.4 cmdline.h File Reference

This graph shows which files directly or indirectly include this file:



Functions

- int **read_command_line** (int, char **)

7.4.1 Function Documentation

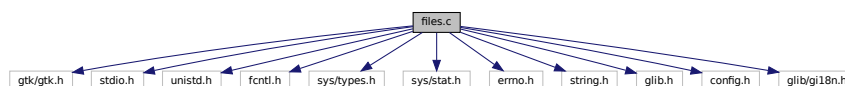
7.4.1.1 read_command_line()

```
int read_command_line (  
    int ,  
    char ** )
```

7.5 files.c File Reference

```
#include <gtk/gtk.h>  
#include <stdio.h>  
#include <unistd.h>  
#include <fcntl.h>  
#include <sys/types.h>  
#include <sys/stat.h>  
#include <errno.h>  
#include <string.h>  
#include <glib.h>  
#include <config.h>  
#include <glib/glib.h>
```

Include dependency graph for files.c:



Variables

- char * **default_filename** = NULL

7.5.1 Variable Documentation

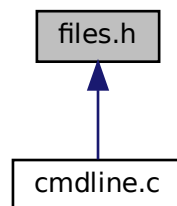
7.5.1.1 default_filename

```
char* default_filename = NULL
```

Referenced by **read_command_line()**.

7.6 files.h File Reference

This graph shows which files directly or indirectly include this file:



Functions

- void **send_raw_file** (GAction *action, gpointer data)
- void **save_raw_file** (GAction *action, gpointer data)
- void **add_input** (void)

Variables

- gboolean **waiting_for_char**
- char * **default_filename**

7.6.1 Function Documentation

7.6.1.1 add_input()

```
void add_input (
    void )
```

7.6.1.2 save_raw_file()

```
void save_raw_file (
    GAction * action,
    gpointer data )
```

7.6.1.3 send_raw_file()

```
void send_raw_file (
    GAction * action,
    gpointer data )
```

7.6.2 Variable Documentation

7.6.2.1 default_filename

```
char* default_filename [extern]
```

Referenced by **read_command_line()**.

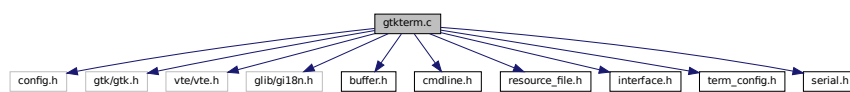
7.6.2.2 waiting_for_char

```
gboolean waiting_for_char [extern]
```

7.7 gtkterm.c File Reference

```
#include "config.h"
#include <gtk/gtk.h>
#include <vte/vte.h>
#include <glib/glib.h>
#include "buffer.h"
#include "cmdline.h"
#include "resource_file.h"
#include "interface.h"
#include "term_config.h"
#include "serial.h"
```

Include dependency graph for gtkterm.c:



Classes

- struct **GtkTermWindow**

Typedefs

- typedef GtkApplication **GtkTerm**
- typedef GtkApplicationClass **GtkTermClass**
- typedef GtkApplicationWindowClass **GtkTermWindowClass**

Functions

- void **set_window_title** (**GtkTermWindow** *)
- int **main** (int argc, char *argv[])

7.7.1 Typedef Documentation

7.7.1.1 GtkTerm

```
typedef GtkApplication GtkTerm
```

7.7.1.2 GtkTermClass

```
typedef GtkApplicationClass  GtkTermClass
```

7.7.1.3 GtkTermWindowClass

```
typedef GtkApplicationWindowClass  GtkTermWindowClass
```

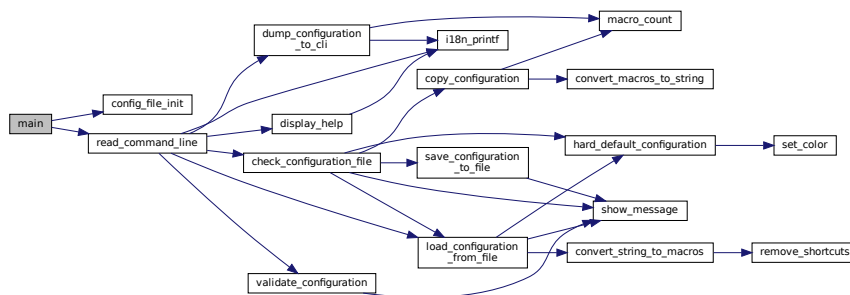
7.7.2 Function Documentation

7.7.2.1 main()

```
int main (
    int argc,
    char * argv[] )
```

References **config_file_init()**, and **read_command_line()**.

Here is the call graph for this function:



7.7.2.2 set_window_title()

```
void set_window_title (
    GtkTermWindow * window )
```

References **get_port_string()**.

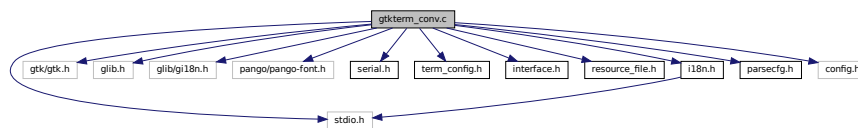
Here is the call graph for this function:



7.8 gtkterm_conv.c File Reference

```
#include <stdio.h>
#include <gtk/gtk.h>
#include <glib.h>
#include <glib/gi18n.h>
#include <pango/pango-font.h>
#include "serial.h"
#include "term_config.h"
#include "interface.h"
#include "resource_file.h"
#include "i18n.h"
#include "parsecfg.h"
#include <config.h>
```

Include dependency graph for gtkterm_conv.c:



Functions

- int **load_old_configuration_from_file** (int)
load old config file with parsecfg Because we convert all sections we can walk trough all section numbers
- void **show_message** (char *msg, int type)
This is the cli version of the one in gtkterm.
- int **main** (int argc, char **argv)

Variables

- **cfgStruct** **cfg** []
- **display_config_t** **term_conf**
Define external variables here configuration for terminal window and serial port.
- **port_config_t** **port_conf**

7.8.1 Function Documentation

7.8.1.1 load_old_configuration_from_file()

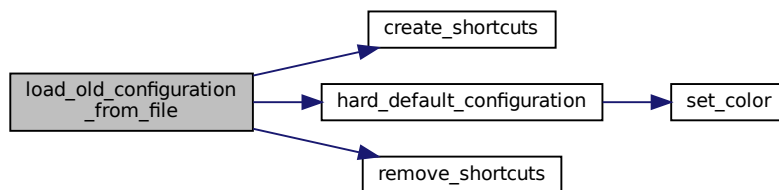
```
int load_old_configuration_from_file (
    int section_nr )
```

load old config file with parsecfg Because we convert all sections we can walk trough all section numbers

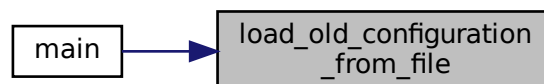
References `macro_t::action`, `background_alpha`, `background_blue`, `display_config_t::background_color`, `background_green`, `background_red`, `bits`, `port_config_t::bits`, `block_cursor`, `display_config_t::block_cursor`, `display_config_t::char_queue`, `columns`, `display_config_t::columns`, `create_shortcuts()`, `crlfauto`, `display_config_t::crlfauto`, `display_config_t::delay`, `echo`, `display_config_t::echo`, `flow`, `port_config_t::flow_control`, `font`, `display_config_t::font`, `foreground_alpha`, `foreground_blue`, `display_config_t::foreground_color`, `foreground_green`, `foreground_red`, `hard_default_configuration()`, `macro_list`, `macros`, `cfgList_tag::next`, `parity`, `port_config_t::parity`, `port`, `port_config_t::port`, `port_config_t::rs485_rts_time_after_transmit`, `port_config_t::rs485_rts_time_before_transmit`, `rts_time_after_tx`, `rts_time_before_tx`, `scrollback`, `display_config_t::scrollback`, `macro_t::shortcut`, `speed`, `port_config_t::speed`, `stopbits`, `port_config_t::stops`, `cfgList_tag::str`, `term_conf`, `timestamp`, `display_config_t::timestamp`, `visual_bell`, `display_config_t::visual_bell`, `wait_char`, and `wait_delay`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.8.1.2 main()

```
int main (
    int argc,
    char ** argv )
```

Initialize for localization

Check if the file exists

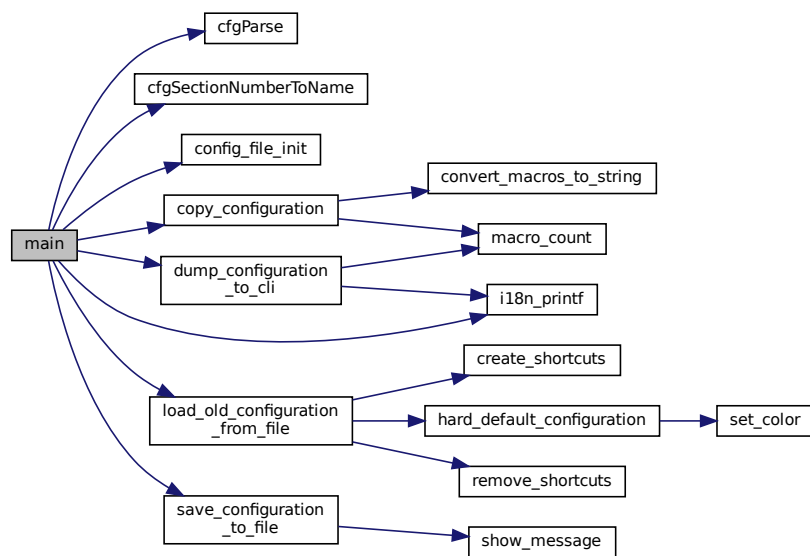
load old config file with parsecfg

Copy the section into the '2.0' structure and save it

Dump all sections to cli

References `cfg`, `CFG_INI`, `cfgParse()`, `cfgSectionNumberToName()`, `config_file`, `config_file_init()`, `copy_configuration()`, `dump_configuration_to_cli()`, `i18n_printf()`, `load_old_configuration_from_file()`, and `save_configuration_to_file()`.

Here is the call graph for this function:



7.8.1.3 show_message()

```
void show_message (
    char * msg,
    int type )
```

This is the cli version of the one in gtkterm.

References `i18n_printf()`.

Here is the call graph for this function:



7.8.2 Variable Documentation

7.8.2.1 `cfg`

```
cfgStruct cfg[] [extern]
```

Referenced by `main()`.

7.8.2.2 `port_conf`

```
port_config_t port_conf
```

7.8.2.3 `term_conf`

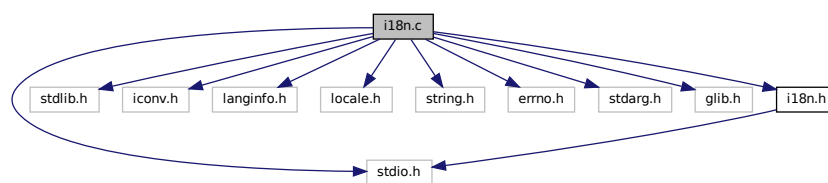
```
display_config_t term_conf
```

Define external variables here configuration for terminal window and serial port.

7.9 i18n.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <iconv.h>
#include <langinfo.h>
#include <locale.h>
#include <string.h>
#include <errno.h>
#include <stdarg.h>
#include <glib.h>
#include "i18n.h"
```

Include dependency graph for i18n.c:



Functions

- int **i18n_printf** (const char *format,...)
- int **i18n_fprintf** (FILE *stream, const char *format,...)
- void **i18n_perror** (const char *s)
- char * **strerror_utf8** (int errornum)

7.9.1 Function Documentation

7.9.1.1 i18n_fprintf()

```
int i18n_fprintf (
    FILE * stream,
    const char * format,
    ... )
```

7.9.1.2 i18n_perror()

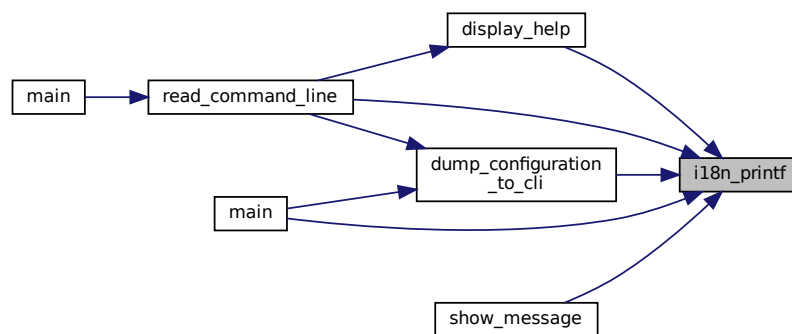
```
void i18n_perror (
    const char * s )
```

7.9.1.3 i18n_printf()

```
int i18n_printf (
    const char * format,
    ... )
```

Referenced by **display_help()**, **dump_configuration_to_cli()**, **main()**, **read_command_line()**, and **show_message()**.

Here is the caller graph for this function:



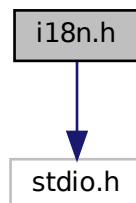
7.9.1.4 strerror_utf8()

```
char* strerror_utf8 (
    int errornum )
```

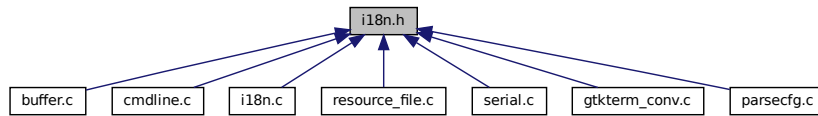
7.10 i18n.h File Reference

```
#include <stdio.h>
```

Include dependency graph for `i18n.h`:



This graph shows which files directly or indirectly include this file:



Macros

- `#define I18N_H`

Functions

- `int i18n_printf (const char *,...)`
- `int i18n_fprintf (FILE *, const char *,...)`
- `void i18n_perror (const char *)`
- `char * strerror_utf8 (int)`

7.10.1 Macro Definition Documentation

7.10.1.1 I18N_H

```
#define I18N_H
```

7.10.2 Function Documentation

7.10.2.1 i18n_fprintf()

```
int i18n_fprintf (  
    FILE * stream,  
    const char * format,  
    ... )
```

7.10.2.2 i18n_perror()

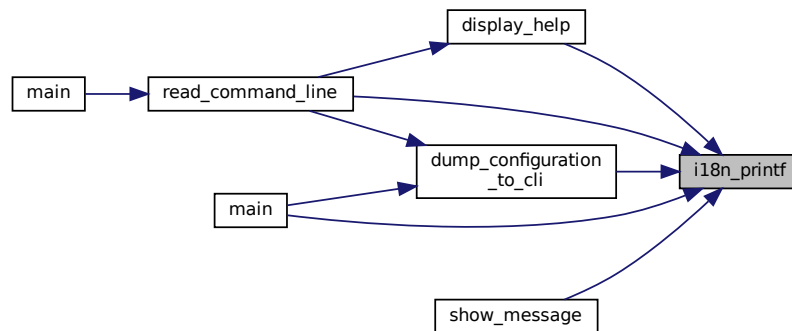
```
void i18n_perror (  
    const char * s )
```

7.10.2.3 i18n_printf()

```
int i18n_printf (
    const char * format,
    ... )
```

Referenced by **display_help()**, **dump_configuration_to_cli()**, **main()**, **read_command_line()**, and **show_message()**.

Here is the caller graph for this function:



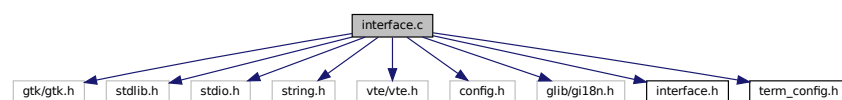
7.10.2.4 strerror_utf8()

```
char* strerror_utf8 (
    int errornum )
```

7.11 interface.c File Reference

```
#include <gtk/gtk.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <vte/vte.h>
#include <config.h>
#include <glib/gi18n.h>
#include "interface.h"
#include "term_config.h"
```

Include dependency graph for interface.c:



Functions

- void **show_message** (char *message, int type_msg)

Variables

- gboolean **timestamp_on** = 0
- struct configuration_port **config**
- int **virt_col_pos** = 0
- GtkWidget * **display** = NULL

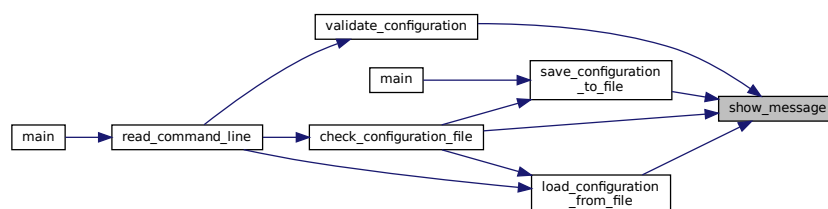
7.11.1 Function Documentation

7.11.1.1 show_message()

```
void show_message (
    char * message,
    int type_msg )
```

Referenced by **check_configuration_file()**, **load_configuration_from_file()**, **save_configuration_to_file()**, and **validate_configuration()**.

Here is the caller graph for this function:



7.11.2 Variable Documentation

7.11.2.1 config

```
struct configuration_port config [extern]
```

7.11.2.2 display

```
GtkWidget* display = NULL
```

7.11.2.3 timestamp_on

```
gboolean timestamp_on = 0
```

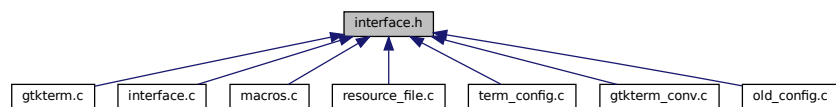
Referenced by `put_chars()`.

7.11.2.4 virt_col_pos

```
int virt_col_pos = 0
```

7.12 interface.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- `#define MSG_WRN 0`
- `#define MSG_ERR 1`
- `#define ASCII_VIEW 0`
- `#define HEXADECIMAL_VIEW 1`

Functions

- void `show_message` (char *, int)
This is the cli version of the one in gtkterm.

Variables

- GtkWidget * `Text`
- GtkWidget * `display`

7.12.1 Macro Definition Documentation

7.12.1.1 ASCII_VIEW

```
#define ASCII_VIEW 0
```

7.12.1.2 HEXADECIMAL_VIEW

```
#define HEXADECIMAL_VIEW 1
```

7.12.1.3 MSG_ERR

```
#define MSG_ERR 1
```

7.12.1.4 MSG_WRN

```
#define MSG_WRN 0
```

7.12.2 Function Documentation

7.12.2.1 show_message()

```
void show_message (
    char * message,
    int type_msg )
```

This is the cli version of the one in gtkterm.

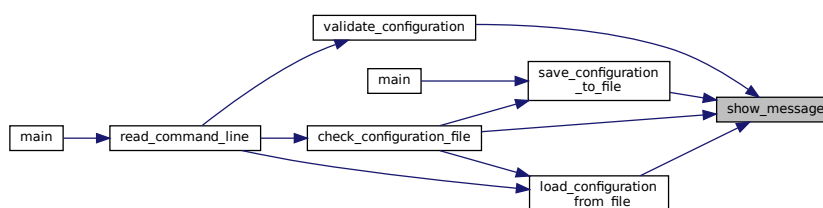
References **i18n_printf()**.

Referenced by **check_configuration_file()**, **load_configuration_from_file()**, **save_configuration_to_file()**, and **validate_configuration()**.

Here is the call graph for this function:



Here is the caller graph for this function:



7.12.3 Variable Documentation

7.12.3.1 display

```
GtkWidget* display [extern]
```

7.12.3.2 Text

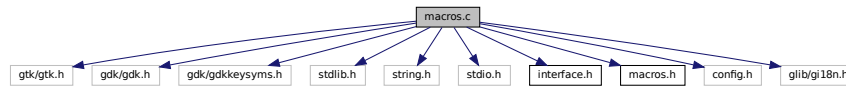
```
GtkWidget* Text [extern]
```

7.13 macros.c File Reference

```

#include <gtk/gtk.h>
#include <gdk/gdk.h>
#include <gdk/gdkkeysyms.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include "interface.h"
  
```

```
#include "macros.h"
#include <config.h>
#include <glib/glib.h>
Include dependency graph for macros.c:
```



Enumerations

- enum { **COLUMN_SHORTCUT** , **COLUMN_ACTION** , **NUM_COLUMNS** }

Functions

- int **macro_count** ()
- void **convert_string_to_macros** (char **string_list, int size)
Convert the array of strings to macros.
- int **convert_macros_to_string** (char **string_list)
Convert the in memory macros to an array of strings for storage in file.
- macro_t** * **get_shortcuts** (int *size)
- void **remove_shortcuts** (void)

Variables

- macro_t** * **macros** = NULL
- int **nr_of_macros** = 0

7.13.1 Enumeration Type Documentation

7.13.1.1 anonymous enum

anonymous enum

Enumerator

COLUMN_SHORTCUT	
COLUMN_ACTION	
NUM_COLUMNS	

7.13.2 Function Documentation

7.13.2.1 convert_macros_to_string()

```
int convert_macros_to_string (
    char ** string_list )
```

Convert the in memory macros to an array of strings for storage in file.

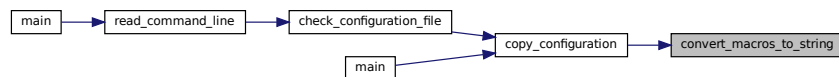
Must be NULL terminated

Number of strings is 2x the macros (shortcut and action)

References **macro_t::action**, **macros**, **nr_of_macros**, and **macro_t::shortcut**.

Referenced by **copy_configuration()**.

Here is the caller graph for this function:



7.13.2.2 convert_string_to_macros()

```
void convert_string_to_macros (
    char ** string_list,
    int size )
```

Convert the array of strings to macros.

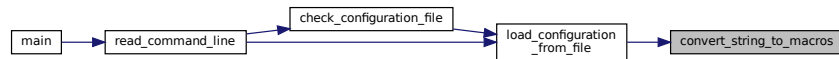
References **macro_t::action**, **macros**, **nr_of_macros**, **remove_shortcuts()**, and **macro_t::shortcut**.

Referenced by **load_configuration_from_file()**.

Here is the call graph for this function:



Here is the caller graph for this function:



7.13.2.3 get_shortcuts()

```

macro_t* get_shortcuts (
    int * size )
  
```

References **macros**.

7.13.2.4 macro_count()

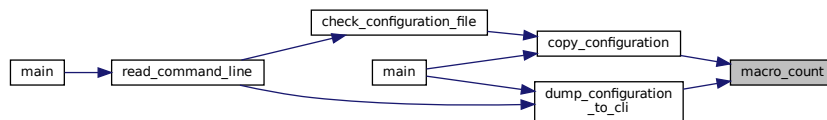
```

int macro_count ( )
  
```

References **nr_of_macros**.

Referenced by **copy_configuration()**, and **dump_configuration_to_cli()**.

Here is the caller graph for this function:



7.13.2.5 remove_shortcuts()

```

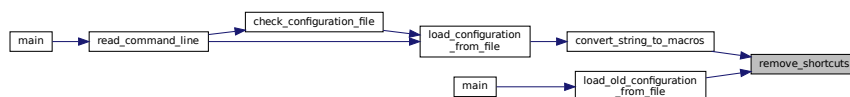
void remove_shortcuts (
    void )
  
```

Clean up all macros

References **macros**.

Referenced by **convert_string_to_macros()**, and **load_old_configuration_from_file()**.

Here is the caller graph for this function:



7.13.3 Variable Documentation

7.13.3.1 macros

```
macro_t* macros = NULL
```

Referenced by `convert_macros_to_string()`, `convert_string_to_macros()`, `create_shortcuts()`, `dump_configuration_to_cli()`, `get_shortcuts()`, `load_old_configuration_from_file()`, and `remove_shortcuts()`.

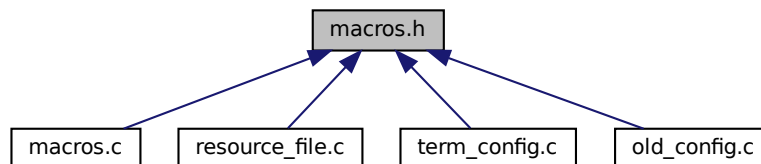
7.13.3.2 nr_of_macros

```
int nr_of_macros = 0
```

Referenced by `convert_macros_to_string()`, `convert_string_to_macros()`, `create_shortcuts()`, and `macro_count()`.

7.14 macros.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- struct **macro_t**
Define macro structure type.

Functions

- void **remove_shortcuts** (void)
Remove shortcuts from accel_group and free memory.
- void **add_shortcuts** (void)
- **macro_t** * **get_shortcuts** (gint *)
Convert the array of strings to macros.
- void **convert_string_to_macros** (char **, int)
Convert the in memory macros to an array of strings for storage in file.
- int **convert_macros_to_string** (char **)
- int **macro_count** ()

Variables

- `macro_t * macros`

7.14.1 Function Documentation

7.14.1.1 `add_shortcuts()`

```
void add_shortcuts (
    void )
```

Remove shortcuts from `accel_group` and free memory.

7.14.1.2 `convert_macros_to_string()`

```
int convert_macros_to_string (
    char ** string_list )
```

Convert the in memory macros to an array of strings for storage in file.

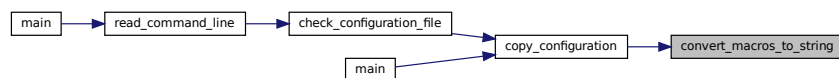
Must be NULL terminated

Number of strings is 2x the macros (shortcut and action)

References `macro_t::action`, `macros`, `nr_of_macros`, and `macro_t::shortcut`.

Referenced by `copy_configuration()`.

Here is the caller graph for this function:



7.14.1.3 convert_string_to_macros()

```
void convert_string_to_macros (
    char ** string_list,
    int size )
```

Convert the array of strings to macros.

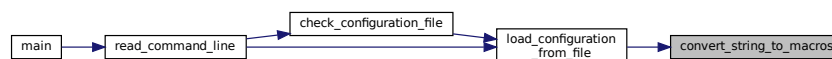
References **macro_t::action**, **macros**, **nr_of_macros**, **remove_shortcuts()**, and **macro_t::shortcut**.

Referenced by **load_configuration_from_file()**.

Here is the call graph for this function:



Here is the caller graph for this function:



7.14.1.4 get_shortcuts()

```
macro_t* get_shortcuts (
    gint * )
```

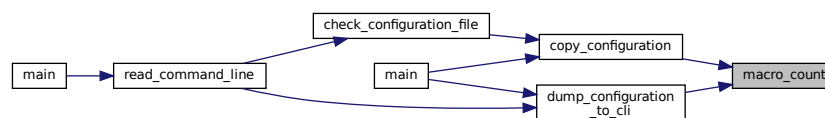
7.14.1.5 macro_count()

```
int macro_count ( )
```

References **nr_of_macros**.

Referenced by **copy_configuration()**, and **dump_configuration_to_cli()**.

Here is the caller graph for this function:



7.14.1.6 remove_shortcuts()

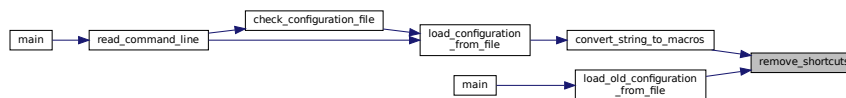
```
void remove_shortcuts (
    void )
```

Clean up all macros

References **macros**.

Referenced by **convert_string_to_macros()**, and **load_old_configuration_from_file()**.

Here is the caller graph for this function:



7.14.2 Variable Documentation

7.14.2.1 macros

```
macro_t* macros [extern]
```

Referenced by **convert_macros_to_string()**, **convert_string_to_macros()**, **create_shortcuts()**, **dump_configuration_to_cli()**, **get_shortcuts()**, **load_old_configuration_from_file()**, and **remove_shortcuts()**.

7.15 meson_post_install.py File Reference

Namespaces

- **meson_post_install**

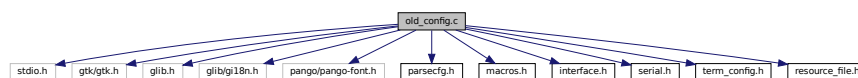
Variables

- **meson_post_install.install_prefix** = `os.environ['MESON_INSTALL_PREFIX']`
- **meson_post_install.schemadir** = `os.path.join(install_prefix, 'share', 'glib-2.0', 'schemas')`

7.16 old_config.c File Reference

```
#include <stdio.h>
#include <gtk/gtk.h>
#include <glib.h>
#include <glib/glib.h>
#include <pango/pango-font.h>
#include "parsecfg.h"
#include "macros.h"
#include "interface.h"
#include "serial.h"
#include "term_config.h"
#include "resource_file.h"
```

Include dependency graph for old_config.c:



Functions

- void **create_shortcuts** (**macro_t** *macro, int size)
Proberbly we only need it for old_config So can be removed from gtkterm.
- int **load_old_configuration_from_file** (int section_nr)
load old config file with parsecfg Because we convert all sections we can walk trough all section numbers

Variables

- int **nr_of_macros**
- char ** **port**
- int * **speed**
- int * **bits**
- int * **stopbits**
- char ** **parity**
- char ** **flow**
- int * **wait_delay**
- int * **wait_char**
- int * **rts_time_before_tx**
- int * **rts_time_after_tx**
- int * **echo**
- int * **crlfauto**
- int * **timestamp**
- **cfgList** ** **macro_list** = NULL
- char ** **font**
- int * **block_cursor**
- int * **rows**
- int * **columns**
- int * **scrollback**
- int * **visual_bell**
- float * **foreground_red**

- float * **foreground_blue**
- float * **foreground_green**
- float * **foreground_alpha**
- float * **background_red**
- float * **background_blue**
- float * **background_green**
- float * **background_alpha**
- **cfgStruct** **cfg** []

7.16.1 Function Documentation

7.16.1.1 create_shortcuts()

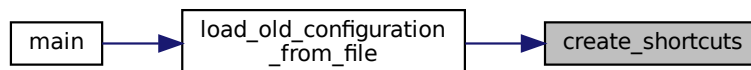
```
void create_shortcuts (
    macro_t * macro,
    int size )
```

Proberbly we only need it for old_config So can be removed from gtkterm.

References **macro_t::action**, **macros**, **nr_of_macros**, and **macro_t::shortcut**.

Referenced by **load_old_configuration_from_file()**.

Here is the caller graph for this function:



7.16.1.2 load_old_configuration_from_file()

```
int load_old_configuration_from_file (
    int section_nr )
```

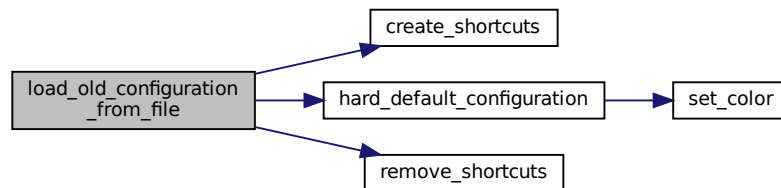
load old config file with parsecfg Because we convert all sections we can walk trough all section numbers

References **macro_t::action**, **background_alpha**, **background_blue**, **display_config_t::background_color**, **background_green**, **background_red**, **bits**, **port_config_t::bits**, **block_cursor**, **display_config_t::block_↵_cursor**, **display_config_t::char_queue**, **columns**, **display_config_t::columns**, **create_shortcuts()**, **crlfauto**, **display_config_t::crlfauto**, **display_config_t::delay**, **echo**, **display_config_t::echo**, **flow**, **port_↵_config_t::flow_control**, **font**, **display_config_t::font**, **foreground_alpha**, **foreground_blue**, **display_↵_config_t::foreground_color**, **foreground_green**, **foreground_red**, **hard_default_configuration()**, **macro_↵_list**, **macros**, **cfgList_tag::next**, **parity**, **port_config_t::parity**, **port**, **port_config_t::port**, **port_conf**,

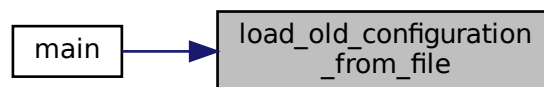
`remove_shortcuts()`, `rows`, `display_config_t::rows`, `port_config_t::rs485_rts_time_after_transmit`, `port_config_t::rs485_rts_time_before_transmit`, `rts_time_after_tx`, `rts_time_before_tx`, `scrollback`, `display_config_t::scrollback`, `macro_t::shortcut`, `speed`, `port_config_t::speed`, `stopbits`, `port_config_t::stops`, `cfgList_tag::str`, `term_conf`, `timestamp`, `display_config_t::timestamp`, `visual_bell`, `display_config_t::visual_bell`, `wait_char`, and `wait_delay`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.16.2 Variable Documentation

7.16.2.1 background_alpha

`float* background_alpha`

Referenced by `load_old_configuration_from_file()`.

7.16.2.2 background_blue

`float* background_blue`

Referenced by `load_old_configuration_from_file()`.

7.16.2.3 background_green

`float* background_green`

Referenced by `load_old_configuration_from_file()`.

7.16.2.4 background_red

`float* background_red`

Referenced by `load_old_configuration_from_file()`.

7.16.2.5 bits

`int* bits`

Referenced by `load_old_configuration_from_file()`.

7.16.2.6 block_cursor

`int* block_cursor`

Referenced by `load_old_configuration_from_file()`.

7.16.2.7 cfg

`cfgStruct cfg[]`

Referenced by `main()`.

7.16.2.8 columns

`int* columns`

Referenced by `load_old_configuration_from_file()`.

7.16.2.9 crlfauto

`int* crlfauto`

Referenced by `load_old_configuration_from_file()`.

7.16.2.10 echo

`int* echo`

Referenced by `load_old_configuration_from_file()`.

7.16.2.11 flow

`char** flow`

Referenced by `load_old_configuration_from_file()`.

7.16.2.12 font

`char** font`

Referenced by `load_old_configuration_from_file()`.

7.16.2.13 foreground_alpha

`float* foreground_alpha`

Referenced by `load_old_configuration_from_file()`.

7.16.2.14 foreground_blue

`float* foreground_blue`

Referenced by `load_old_configuration_from_file()`.

7.16.2.15 foreground_green

```
float* foreground_green
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.16 foreground_red

```
float* foreground_red
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.17 macro_list

```
cfgList** macro_list = NULL
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.18 nr_of_macros

```
int nr_of_macros [extern]
```

Referenced by `convert_macros_to_string()`, `convert_string_to_macros()`, `create_shortcuts()`, and `macro↵_count()`.

7.16.2.19 parity

```
char** parity
```

Referenced by `get_port_string()`, and `load_old_configuration_from_file()`.

7.16.2.20 port

```
char** port
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.21 rows

```
int* rows
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.22 rts_time_after_tx

```
int* rts_time_after_tx
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.23 rts_time_before_tx

```
int* rts_time_before_tx
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.24 scrollbar

```
int* scrollbar
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.25 speed

```
int* speed
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.26 stopbits

```
int* stopbits
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.27 timestamp

```
int* timestamp
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.28 visual_bell

```
int* visual_bell
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.29 wait_char

```
int* wait_char
```

Referenced by `load_old_configuration_from_file()`.

7.16.2.30 wait_delay

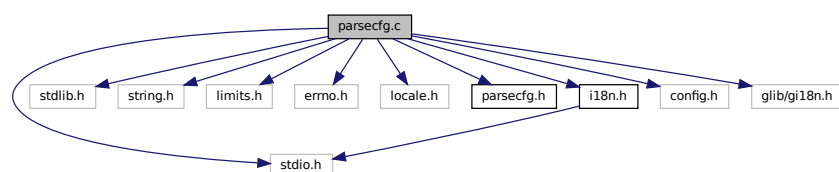
```
int* wait_delay
```

Referenced by `load_old_configuration_from_file()`.

7.17 parsecfg.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <limits.h>
#include <errno.h>
#include <locale.h>
#include "parsecfg.h"
#include "i18n.h"
#include <config.h>
#include <glib/gi18n.h>
```

Include dependency graph for `parsecfg.c`:



Functions

- void **cfgSetFatalFunc** (void(*)(f)(**cfgErrorCode**, const char *, int, const char *))
- int **cfgParse** (const char *file, **cfgStruct** **cfg**[], **cfgFileType** type)
- int **cfgDump** (const char *file, **cfgStruct** **cfg**[], **cfgFileType** type, int max_section)
- int **fetchVarFromCfgFile** (const char *file, char *parameter_name, void *result_value, **cfgValueType** value_type, **cfgFileType** file_type, int section_num, const char *section_name)
- int **cfgSectionNameToNumber** (const char *name)
- char * **cfgSectionNumberToName** (int num)
- int **cfgAllocForNewSection** (**cfgStruct** **cfg**[], const char *name)
- int **cfgStoreValue** (**cfgStruct** **cfg**[], const char *parameter, const char *value, **cfgFileType** type, int section)

7.17.1 Function Documentation

7.17.1.1 **cfgAllocForNewSection()**

```
int cfgAllocForNewSection (
    cfgStruct cfg[],
    const char * name )
```

7.17.1.2 **cfgDump()**

```
int cfgDump (
    const char * file,
    cfgStruct cfg[],
    cfgFileType type,
    int max_section )
```

7.17.1.3 **cfgParse()**

```
int cfgParse (
    const char * file,
    cfgStruct cfg[],
    cfgFileType type )
```

Referenced by **main()**.

Here is the caller graph for this function:



7.17.1.4 `cfgSectionNameToNumber()`

```
int cfgSectionNameToNumber (
    const char * name )
```

7.17.1.5 `cfgSectionNumberToName()`

```
char* cfgSectionNumberToName (
    int num )
```

Referenced by `main()`.

Here is the caller graph for this function:



7.17.1.6 `cfgSetFatalFunc()`

```
void cfgSetFatalFunc (
    void(*) ( cfgErrorCode, const char *, int, const char *) f )
```

7.17.1.7 `cfgStoreValue()`

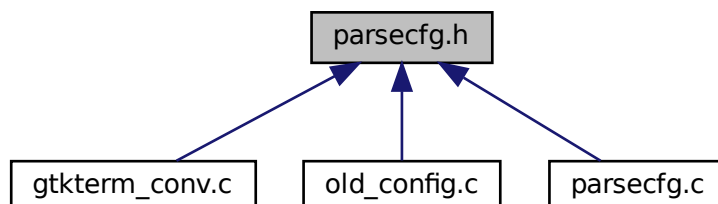
```
int cfgStoreValue (
    cfgStruct cfg[],
    const char * parameter,
    const char * value,
    cfgFileType type,
    int section )
```

7.17.1.8 fetchVarFromCfgFile()

```
int fetchVarFromCfgFile (
    const char * file,
    char * parameter_name,
    void * result_value,
    cfgValueType value_type,
    cfgFileType file_type,
    int section_num,
    const char * section_name )
```

7.18 parsecfg.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- struct **cfgStruct**
- struct **cfgList_tag**

Macros

- #define **PARSECFG_VERSION** "3.6.7"

Typedefs

- typedef struct **cfgList_tag** **cfgList**

Enumerations

- enum **cfgErrorCode** {
CFG_NO_ERROR, **CFG_OPEN_FAIL**, **CFG_CREATE_FAIL**, **CFG_SYNTAX_ERROR**,
CFG_WRONG_PARAMETER, **CFG_INTERNAL_ERROR**, **CFG_INVALID_NUMBER**, **CFG_OUT_OF_**↵
_RANGE,
CFG_MEM_ALLOC_FAIL, **CFG_BOOL_ERROR**, **CFG_USED_SECTION**, **CFG_NO_CLOSING_**↵
BRACE,
CFG_JUST_RETURN_WITHOUT_MSG }
- enum **cfgFileType** { **CFG_SIMPLE**, **CFG_INI** }
- enum **cfgValueType** {
CFG_END, **CFG_BOOL**, **CFG_STRING**, **CFG_INT**,
CFG_UINT, **CFG_LONG**, **CFG_ULONG**, **CFG_STRING_LIST**,
CFG_FLOAT, **CFG_DOUBLE** }
- enum **cfgKeywordValue** { **CFG_PARAMETER**, **CFG_VALUE**, **CFG_SECTION** }
- enum **cfgQuote** { **CFG_NO_QUOTE**, **CFG_SINGLE_QUOTE**, **CFG_DOUBLE_QUOTE** }

Functions

- void **cfgSetFatalFunc** (void(*f)(**cfgErrorCode**, const char *, int, const char *))
- int **cfgParse** (const char *file, **cfgStruct** cfg[], **cfgFileType** type)
- int **cfgDump** (const char *file, **cfgStruct** cfg[], **cfgFileType** type, int max_section)
- int **fetchVarFromCfgFile** (const char *file, char *parameter_name, void *result_value, **cfgValueType** value_type, **cfgFileType** file_type, int section_num, const char *section_name)
- int **cfgSectionNameToNumber** (const char *name)
- char * **cfgSectionNumberToName** (int num)
- int **cfgAllocForNewSection** (**cfgStruct** cfg[], const char *name)
- int **cfgStoreValue** (**cfgStruct** cfg[], const char *parameter, const char *value, **cfgFileType** type, int section)

7.18.1 Macro Definition Documentation

7.18.1.1 PARSECFG_VERSION

```
#define PARSECFG_VERSION "3.6.7"
```

7.18.2 Typedef Documentation

7.18.2.1 cfgList

```
typedef struct cfgList_tag cfgList
```

7.18.3 Enumeration Type Documentation

7.18.3.1 cfgErrorCode

```
enum cfgErrorCode
```

Enumerator

CFG_NO_ERROR	
CFG_OPEN_FAIL	
CFG_CREATE_FAIL	
CFG_SYNTAX_ERROR	
CFG_WRONG_PARAMETER	
CFG_INTERNAL_ERROR	
CFG_INVALID_NUMBER	
CFG_OUT_OF_RANGE	
CFG_MEM_ALLOC_FAIL	
CFG_BOOL_ERROR	
CFG_USED_SECTION	
CFG_NO_CLOSING_BRACE	
CFG_JUST_RETURN_WITHOUT_MSG	

7.18.3.2 cfgFileType

```
enum cfgFileType
```

Enumerator

CFG_SIMPLE	
CFG_INI	

7.18.3.3 cfgKeywordValue

```
enum cfgKeywordValue
```

Enumerator

CFG_PARAMETER	
CFG_VALUE	
CFG_SECTION	

7.18.3.4 cfgQuote

```
enum cfgQuote
```

Enumerator

CFG_NO_QUOTE	
CFG_SINGLE_QUOTE	
CFG_DOUBLE_QUOTE	

7.18.3.5 `cfgValueType`

```
enum cfgValueType
```

Enumerator

CFG_END	
CFG_BOOL	
CFG_STRING	
CFG_INT	
CFG_UINT	
CFG_LONG	
CFG_ULONG	
CFG_STRING_LIST	
CFG_FLOAT	
CFG_DOUBLE	

7.18.4 Function Documentation

7.18.4.1 `cfgAllocForNewSection()`

```
int cfgAllocForNewSection (
    cfgStruct cfg[],
    const char * name )
```

7.18.4.2 `cfgDump()`

```
int cfgDump (
    const char * file,
    cfgStruct cfg[],
    cfgFileType type,
    int max_section )
```

7.18.4.3 cfgParse()

```
int cfgParse (
    const char * file,
    cfgStruct cfg[],
    cfgFileType type )
```

Referenced by **main()**.

Here is the caller graph for this function:



7.18.4.4 cfgSectionNameToNumber()

```
int cfgSectionNameToNumber (
    const char * name )
```

7.18.4.5 cfgSectionNumberToName()

```
char* cfgSectionNumberToName (
    int num )
```

Referenced by **main()**.

Here is the caller graph for this function:



7.18.4.6 cfgSetFatalFunc()

```
void cfgSetFatalFunc (
    void(*) ( cfgErrorCode, const char *, int, const char *) f )
```

7.18.4.7 cfgStoreValue()

```
int cfgStoreValue (
    cfgStruct cfg[],
    const char * parameter,
    const char * value,
    cfgFileType type,
    int section )
```

7.18.4.8 fetchVarFromCfgFile()

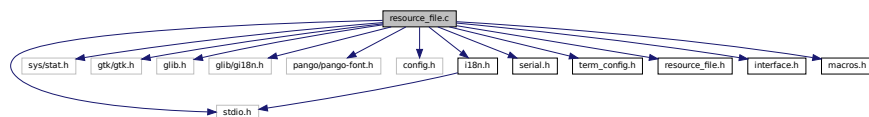
```
int fetchVarFromCfgFile (
    const char * file,
    char * parameter_name,
    void * result_value,
    cfgValueType value_type,
    cfgFileType file_type,
    int section_num,
    const char * section_name )
```

7.19 README.md File Reference

7.20 resource_file.c File Reference

```
#include <stdio.h>
#include <sys/stat.h>
#include <gtk/gtk.h>
#include <glib.h>
#include <glib/glib18n.h>
#include <pango/pango-font.h>
#include <config.h>
#include "il8n.h"
#include "serial.h"
#include "term_config.h"
#include "resource_file.h"
#include "interface.h"
#include "macros.h"
```

Include dependency graph for resource_file.c:



Macros

- `#define CONFIGURATION_FILENAME ".gtktermrc"`

Default configuration filename.

Enumerations

- enum {
`CONF_ITEM_PORT`, `CONF_ITEM_SPEED`, `CONF_ITEM_BITS`, `CONF_ITEM_STOPBITS`,
`CONF_ITEM_PARITY`, `CONF_ITEM_FLOW_CONTROL`, `CONF_ITEM_WAIT_DELAY`, `CONF_ITEM_WAIT_CHAR`,
`CONF_ITEM_RS485_RTS_TIME_BEFORE_TX`, `CONF_ITEM_RS485_RTS_TIME_AFTER_TX`, `CONF_ITEM_MACROS`, `CONF_ITEM_ECHO`,
`CONF_ITEM_CRLF_AUTO`, `CONF_ITEM_DISABLE_PORT_LOCK`, `CONF_ITEM_FONT`, `CONF_ITEM_TERM_SHOW_CURSOR`,
`CONF_ITEM_TERM_ROWS`, `CONF_ITEM_TERM_COLS`, `CONF_ITEM_TERM_SCROLLBACK`,
`CONF_ITEM_TERM_VISUAL_BELL`,
`CONF_ITEM_TERM_FOREGROUND_RED`, `CONF_ITEM_TERM_FOREGROUND_GREEN`, `CONF_ITEM_TERM_FOREGROUND_BLUE`,
`CONF_ITEM_TERM_FOREGROUND_ALPHA`,
`CONF_ITEM_TERM_BACKGROUND_RED`, `CONF_ITEM_TERM_BACKGROUND_GREEN`, `CONF_ITEM_TERM_BACKGROUND_BLUE`,
`CONF_ITEM_TERM_BACKGROUND_ALPHA` }

Define all configuration items which are used in the resource file.

Functions

- void `config_file_init` (void)
- void `dump_configuration_to_cli` (char *section)
- void `save_configuration_to_file` (GKeyFile *config, const char *section)
- int `load_configuration_from_file` (const char *section)
- int `check_configuration_file` (void)
This checks if the configuration file exists.
- void `copy_configuration` (GKeyFile *configrc, const char *section)
Copy the active configuration into <section> of the Key file.
- int `remove_section` (char *cfg_file, char *section)
Remove a section from the file TODO: Perhaps remove because we dont need it.
- void `hard_default_configuration` (void)
Create a new <default> configuration.
- void `validate_configuration` (void)
validate the active configuration
- void `set_color` (GdkRGBA *color, float R, float G, float B, float A)
Convert the colors RGB to internal color scheme.

Variables

- GFile * `config_file`
The key file.
- char `ConfigurationItem` [][32]

7.20.1 Macro Definition Documentation

7.20.1.1 CONFIGURATION_FILENAME

```
#define CONFIGURATION_FILENAME ".gtktermrc"
```

Default configuration filename.

7.20.2 Enumeration Type Documentation

7.20.2.1 anonymous enum

```
anonymous enum
```

Define all configuration items which are used in the resource file.

it is an index to ConfigurationItem.

Enumerator

CONF_ITEM_PORT	
CONF_ITEM_SPEED	
CONF_ITEM_BITS	
CONF_ITEM_STOPBITS	
CONF_ITEM_PARITY	
CONF_ITEM_FLOW_CONTROL	
CONF_ITEM_WAIT_DELAY	
CONF_ITEM_WAIT_CHAR	
CONF_ITEM_RS485_RTS_TIME_BEFORE_TX	
CONF_ITEM_RS485_RTS_TIME_AFTER_TX	
CONF_ITEM_MACROS	
CONF_ITEM_ECHO	
CONF_ITEM_CRLF_AUTO	
CONF_ITEM_DISABLE_PORT_LOCK	
CONF_ITEM_FONT	
CONF_ITEM_TERM_SHOW_CURSOR	
CONF_ITEM_TERM_ROWS	
CONF_ITEM_TERM_COLS	
CONF_ITEM_TERM_SCROLLBACK	
CONF_ITEM_TERM_VISUAL_BELL	
CONF_ITEM_TERM_FOREGROUND_RED	
CONF_ITEM_TERM_FOREGROUND_GREEN	
CONF_ITEM_TERM_FOREGROUND_BLUE	
CONF_ITEM_TERM_FOREGROUND_ALPHA	
CONF_ITEM_TERM_BACKGROUND_RED	
CONF_ITEM_TERM_BACKGROUND_GREEN	
CONF_ITEM_TERM_BACKGROUND_BLUE	
CONF_ITEM_TERM_BACKGROUND_ALPHA	

7.20.3 Function Documentation

7.20.3.1 check_configuration_file()

```
int check_configuration_file (
    void )
```

This checks if the configuration file exists.

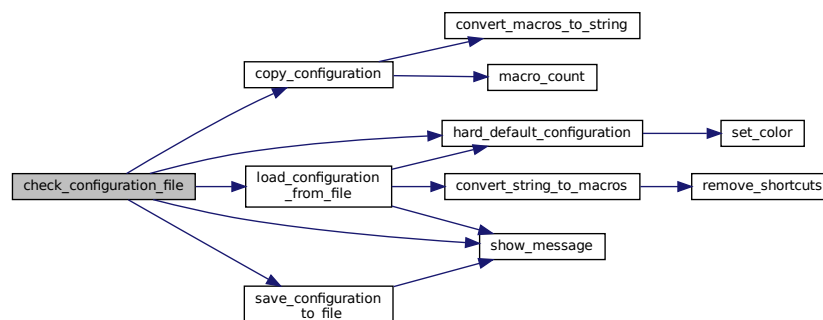
If not it creates a new [default] Put the new default in the key file

And save the config to file

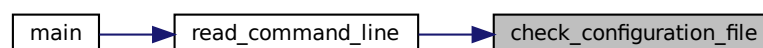
References **config**, **config_file**, **copy_configuration()**, **hard_default_configuration()**, **load_configuration_from_file()**, **MSG_WRN**, **save_configuration_to_file()**, and **show_message()**.

Referenced by **read_command_line()**.

Here is the call graph for this function:



Here is the caller graph for this function:



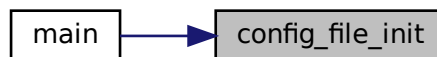
7.20.3.2 config_file_init()

```
void config_file_init (
    void )
```

References **config_file**, and **CONFIGURATION_FILENAME**.

Referenced by **main()**.

Here is the caller graph for this function:



7.20.3.3 copy_configuration()

```
void copy_configuration (
    GKeyFile * configrc,
    const char * section )
```

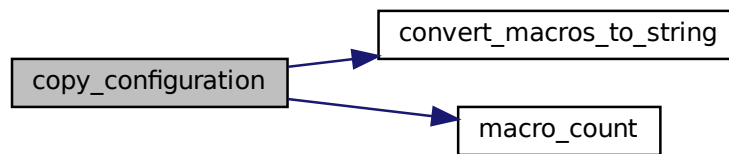
Copy the active configuration into <section> of the Key file.

Macros are an array of strings, so we have to convert it All macros ends up in the string_list

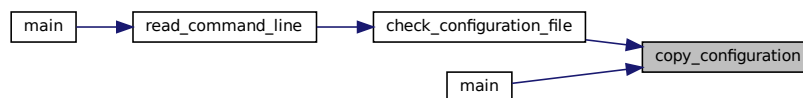
References **display_config_t::background_color**, **port_config_t::bits**, **display_config_t::char_queue**, **display_config_t::columns**, **CONF_ITEM_BITS**, **CONF_ITEM_CRLF_AUTO**, **CONF_ITEM_DISABLE_↵**
_PORT_LOCK, **CONF_ITEM_ECHO**, **CONF_ITEM_FLOW_CONTROL**, **CONF_ITEM_FONT**, **CONF_↵**
ITEM_MACROS, **CONF_ITEM_PARITY**, **CONF_ITEM_PORT**, **CONF_ITEM_RS485_RTS_TIME_AFTER_TX**,
CONF_ITEM_RS485_RTS_TIME_BEFORE_TX, **CONF_ITEM_SPEED**, **CONF_ITEM_STOPBITS**, **CONF_↵**
ITEM_TERM_BACKGROUND_ALPHA, **CONF_ITEM_TERM_BACKGROUND_BLUE**, **CONF_ITEM_TERM_↵**
BACKGROUND_GREEN, **CONF_ITEM_TERM_BACKGROUND_RED**, **CONF_ITEM_TERM_COLS**, **CONF_↵**
ITEM_TERM_FOREGROUND_ALPHA, **CONF_ITEM_TERM_FOREGROUND_BLUE**, **CONF_ITEM_TERM_↵**
FOREGROUND_GREEN, **CONF_ITEM_TERM_FOREGROUND_RED**, **CONF_ITEM_TERM_ROWS**, **CONF_↵**
_ITEM_TERM_SCROLLBACK, **CONF_ITEM_TERM_SHOW_CURSOR**, **CONF_ITEM_TERM_VISUAL_BELL**,
CONF_ITEM_WAIT_CHAR, **CONF_ITEM_WAIT_DELAY**, **ConfigurationItem**, **convert_macros_to_string()**,
display_config_t::crlfauto, **display_config_t::delay**, **port_config_t::disable_port_lock**, **display_config_t_↵**
::echo, **port_config_t::flow_control**, **display_config_t::font**, **display_config_t::foreground_color**, **macro_↵**
_count(), **port_config_t::parity**, **port_config_t::port**, **port_conf**, **display_config_t::rows**, **port_config_↵**
_t::rs485_rts_time_after_transmit, **port_config_t::rs485_rts_time_before_transmit**, **display_config_t_↵**
::scrollback, **display_config_t::show_cursor**, **port_config_t::speed**, **port_config_t::stops**, **term_conf**, and
display_config_t::visual_bell.

Referenced by **check_configuration_file()**, and **main()**.

Here is the call graph for this function:



Here is the caller graph for this function:



7.20.3.4 dump_configuration_to_cli()

```
void dump_configuration_to_cli (
    char * section )
```

Print the serial port items

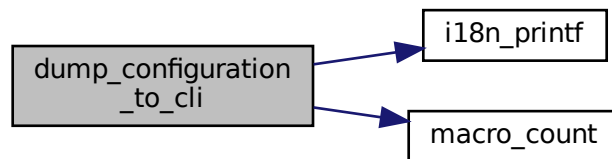
Print the terminal items

... and the macro's

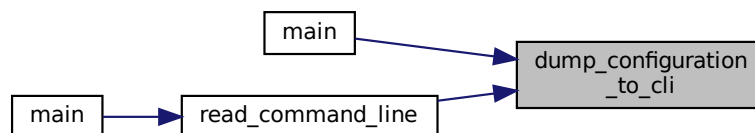
References `display_config_t::background_color`, `port_config_t::bits`, `display_config_t::block_cursor`, `display_config_t::char_queue`, `display_config_t::columns`, `display_config_t::crlfauto`, `display_config_t::delay`, `port_config_t::disable_port_lock`, `display_config_t::echo`, `port_config_t::flow_control`, `display_config_t::font`, `display_config_t::foreground_color`, `i18n_printf()`, `macro_count()`, `macros`, `port_config_t::parity`, `port_config_t::port`, `port_conf`, `display_config_t::rows`, `port_config_t::rs485_rts_time_after_transmit`, `port_config_t::rs485_rts_time_before_transmit`, `display_config_t::scrollback`, `display_config_t::show_cursor`, `port_config_t::speed`, `port_config_t::stops`, `term_conf`, `display_config_t::timestamp`, and `display_config_t::visual_bell`.

Referenced by `main()`, and `read_command_line()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.20.3.5 hard_default_configuration()

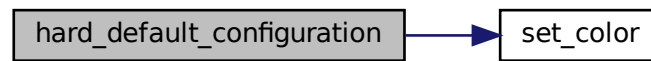
```
void hard_default_configuration (
    void )
```

Create a new <default> configuration.

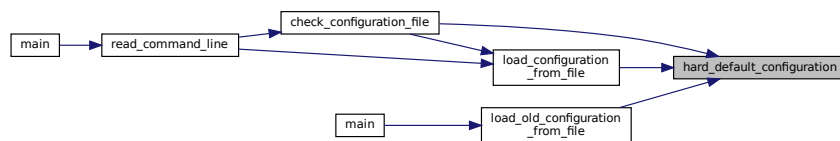
References `display_config_t::background_color`, `port_config_t::bits`, `display_config_t::block_cursor`, `display_config_t::char_queue`, `display_config_t::columns`, `display_config_t::crlfauto`, `DEFAULT_BITS`, `DEFAULT_CHAR`, `DEFAULT_DELAY`, `DEFAULT_DELAY_RS485`, `DEFAULT_ECHO`, `DEFAULT_FLOW`, `DEFAULT_FONT`, `DEFAULT_PARITY`, `DEFAULT_PORT`, `DEFAULT_SCROLLBACK`, `DEFAULT_SPEED`, `DEFAULT_STOP`, `display_config_t::delay`, `port_config_t::disable_port_lock`, `display_config_t::echo`, `port_config_t::flow_control`, `display_config_t::font`, `display_config_t::foreground_color`, `port_config_t::parity`, `port_config_t::port`, `port_conf`, `display_config_t::rows`, `port_config_t::rs485_rts_time_after_transmit`, `port_config_t::rs485_rts_time_before_transmit`, `display_config_t::scrollback`, `set_color()`, `display_config_t::show_cursor`, `port_config_t::speed`, `port_config_t::stops`, `term_conf`, `display_config_t::timestamp`, and `display_config_t::visual_bell`.

Referenced by `check_configuration_file()`, `load_configuration_from_file()`, and `load_old_configuration_from_file()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.20.3.6 load_configuration_from_file()

```
int load_configuration_from_file (
    const char * section )
```

Load the key file Note: all sections are loaded into memory.

Check if the <section> exists in the key file.

First initialize with a default structure. Not really needed.

The Font is a Pango structure. This only can be added to a terminal So we have to convert it.

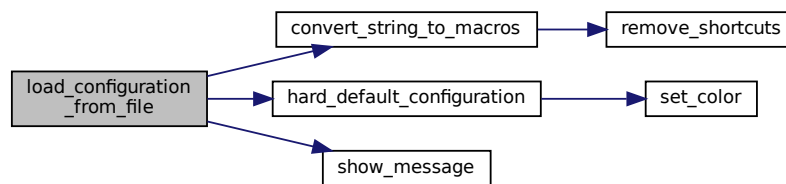
Convert the stringlist to macros. Existing shortcuts will be delete from convert_string_to_macros

References `display_config_t::background_color`, `port_config_t::bits`, `display_config_t::char_queue`, `display_config_t::columns`, `CONF_ITEM_BITS`, `CONF_ITEM_CRLF_AUTO`, `CONF_ITEM_DISABLE`, `_PORT_LOCK`, `CONF_ITEM_ECHO`, `CONF_ITEM_FLOW_CONTROL`, `CONF_ITEM_FONT`, `CONF_ITEM_MACROS`, `CONF_ITEM_PARITY`, `CONF_ITEM_PORT`, `CONF_ITEM_RS485_RTS_TIME_AFTER_TX`, `CONF_ITEM_RS485_RTS_TIME_BEFORE_TX`, `CONF_ITEM_SPEED`, `CONF_ITEM_STOPBITS`, `CONF_ITEM_TERM_BACKGROUND_ALPHA`, `CONF_ITEM_TERM_BACKGROUND_BLUE`, `CONF_ITEM_TERM_BACKGROUND_GREEN`, `CONF_ITEM_TERM_BACKGROUND_RED`, `CONF_ITEM_TERM_COLS`, `CONF_ITEM_TERM_FOREGROUND_ALPHA`, `CONF_ITEM_TERM_FOREGROUND_BLUE`, `CONF_ITEM_TERM_FOREGROUND_GREEN`, `CONF_ITEM_TERM_FOREGROUND_RED`, `CONF_ITEM_TERM_ROWS`, `CONF_ITEM_TERM_SCROLLBACK`, `CONF_ITEM_TERM_SHOW_CURSOR`, `CONF_ITEM_TERM_VISUAL_BELL`,

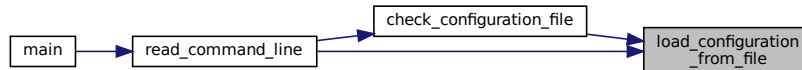
CONF_ITEM_WAIT_CHAR, CONF_ITEM_WAIT_DELAY, config_file, ConfigurationItem, convert_string_to_macros(), display_config_t::crlfauto, display_config_t::delay, port_config_t::disable_port_lock, display_config_t::echo, port_config_t::flow_control, display_config_t::font, display_config_t::foreground_color, hard_default_configuration(), MSG_ERR, port_config_t::parity, port_config_t::port, port_conf, display_config_t::rows, port_config_t::rs485_rts_time_after_transmit, port_config_t::rs485_rts_time_before_transmit, display_config_t::scrollback, display_config_t::show_cursor, show_message(), port_config_t::speed, port_config_t::stops, term_conf, and display_config_t::visual_bell.

Referenced by `check_configuration_file()`, and `read_command_line()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.20.3.7 remove_section()

```

int remove_section (
    char * cfg_file,
    char * section )

```

Remove a section from the file TODO: Perhaps remove because we dont need it.

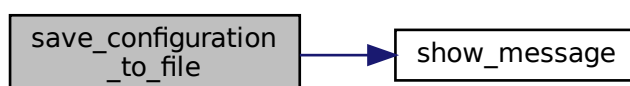
7.20.3.8 save_configuration_to_file()

```
void save_configuration_to_file (  
    GKeyFile * config,  
    const char * section )
```

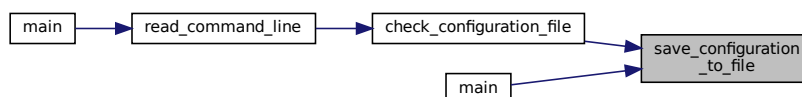
References **config**, **config_file**, **MSG_WRN**, and **show_message()**.

Referenced by **check_configuration_file()**, and **main()**.

Here is the call graph for this function:



Here is the caller graph for this function:



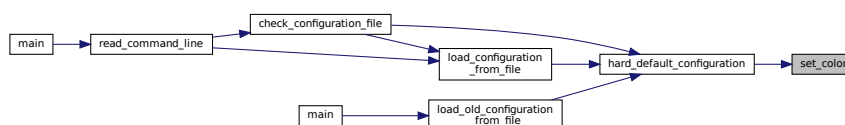
7.20.3.9 set_color()

```
void set_color (  
    GdkRGBA * color,  
    float R,  
    float G,  
    float B,  
    float A )
```

Convert the colors RGB to internal color scheme.

Referenced by **hard_default_configuration()**.

Here is the caller graph for this function:



7.20.3.10 validate_configuration()

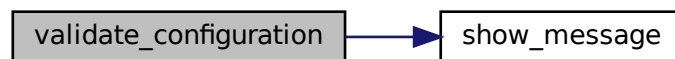
```
void validate_configuration (
    void )
```

validate the active configuration

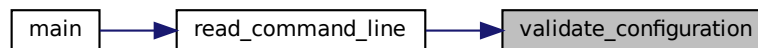
References `port_config_t::bits`, `DEFAULT_BITS`, `DEFAULT_DELAY`, `DEFAULT_FONT`, `DEFAULT_STOP`, `display_config_t::delay`, `display_config_t::font`, `MSG_ERR`, `port_conf`, `show_message()`, `port_config_t::speed`, `port_config_t::stops`, and `term_conf`.

Referenced by `read_command_line()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.20.4 Variable Documentation

7.20.4.1 config_file

```
GFile* config_file
```

The key file.

Referenced by `check_configuration_file()`, `config_file_init()`, `load_configuration_from_file()`, `main()`, and `save_configuration_to_file()`.

7.20.4.2 ConfigurationItem

```
char ConfigurationItem[ ][32]
```

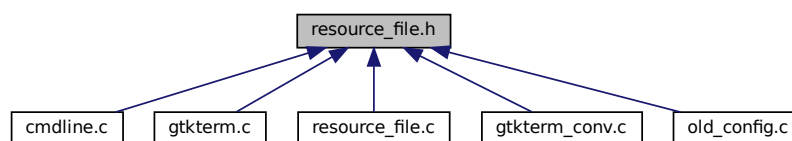
Initial value:

```
= {
    "port",
    "speed",
    "bits",
    "stopbits",
    "parity",
    "flow_control",
    "wait_delay",
    "wait_char",
    "rs485_rts_time_before_tx",
    "rs485_rts_time_after_tx",
    "macros",
    "echo",
    "crlfauto",
    "disable_port_lock",
    "term_font",
    "term_show_cursor",
    "term_rows",
    "term_columns",
    "term_scrollback",
    "term_visual_bell",
    "term_foreground_red",
    "term_foreground_green",
    "term_foreground_blue",
    "term_foreground_alpha",
    "term_background_red",
    "term_background_green",
    "term_background_blue",
    "term_background_alpha"
}
```

Referenced by `copy_configuration()`, and `load_configuration_from_file()`.

7.21 resource_file.h File Reference

This graph shows which files directly or indirectly include this file:



Functions

- void **config_file_init** (void)
- void **save_configuration_to_file** (GKeyFile *, const char *)
- int **load_configuration_from_file** (const char *)
- int **check_configuration_file** ()
- void **dump_configuration_to_cli** (char *)
- void **hard_default_configuration** (void)

This checks if the configuration file exists.

Create a new <default> configuration.

- void **validate_configuration** (void)
validate the active configuration
- void **copy_configuration** (GKeyFile *, const char *)
Copy the active configuration into <section> of the Key file.
- int **remove_section** (char *cfg_file, char *section)
Remove a section from the file TODO: Perhaps remove because we dont need it.
- void **set_color** (GdkRGBA *color, float, float, float, float)
Convert the colors RGB to internal color scheme.

Variables

- GFile * **config_file**
The key file.

7.21.1 Function Documentation

7.21.1.1 check_configuration_file()

```
int check_configuration_file (
    void )
```

This checks if the configuration file exists.

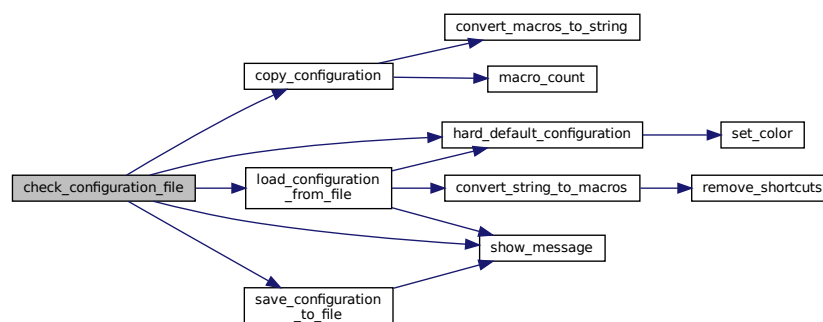
If not it creates a new [default] Put the new default in the key file

And save the config to file

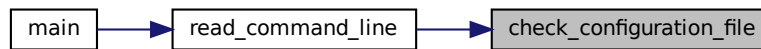
References **config**, **config_file**, **copy_configuration()**, **hard_default_configuration()**, **load_configuration_from_file()**, **MSG_WRN**, **save_configuration_to_file()**, and **show_message()**.

Referenced by **read_command_line()**.

Here is the call graph for this function:



Here is the caller graph for this function:



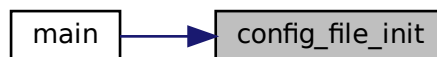
7.21.1.2 config_file_init()

```
void config_file_init (
    void )
```

References **config_file**, and **CONFIGURATION_FILENAME**.

Referenced by **main()**.

Here is the caller graph for this function:



7.21.1.3 copy_configuration()

```
void copy_configuration (
    GKeyFile * configrc,
    const char * section )
```

Copy the active configuration into <section> of the Key file.

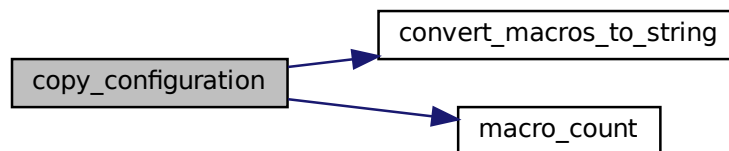
Macros are an array of strings, so we have to convert it All macros ends up in the string_list

References **display_config_t::background_color**, **port_config_t::bits**, **display_config_t::char_queue**, **display_config_t::columns**, **CONF_ITEM_BITS**, **CONF_ITEM_CRLF_AUTO**, **CONF_ITEM_DISABLE_↵**
_PORT_LOCK, **CONF_ITEM_ECHO**, **CONF_ITEM_FLOW_CONTROL**, **CONF_ITEM_FONT**, **CONF_↵**
ITEM_MACROS, **CONF_ITEM_PARITY**, **CONF_ITEM_PORT**, **CONF_ITEM_RS485_RTS_TIME_AFTER_TX**,

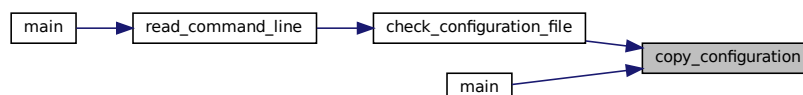
CONF_ITEM_RS485_RTS_TIME_BEFORE_TX, CONF_ITEM_SPEED, CONF_ITEM_STOPBITS, CONF_ITEM_TERM_BACKGROUND_ALPHA, CONF_ITEM_TERM_BACKGROUND_BLUE, CONF_ITEM_TERM_BACKGROUND_GREEN, CONF_ITEM_TERM_BACKGROUND_RED, CONF_ITEM_TERM_COLS, CONF_ITEM_TERM_FOREGROUND_ALPHA, CONF_ITEM_TERM_FOREGROUND_BLUE, CONF_ITEM_TERM_FOREGROUND_GREEN, CONF_ITEM_TERM_FOREGROUND_RED, CONF_ITEM_TERM_ROWS, CONF_ITEM_TERM_SCROLLBACK, CONF_ITEM_TERM_SHOW_CURSOR, CONF_ITEM_TERM_VISUAL_BELL, CONF_ITEM_WAIT_CHAR, CONF_ITEM_WAIT_DELAY, ConfigurationItem, convert_macros_to_string(), display_config_t::crlfauto, display_config_t::delay, port_config_t::disable_port_lock, display_config_t::echo, port_config_t::flow_control, display_config_t::font, display_config_t::foreground_color, macro_count(), port_config_t::parity, port_config_t::port, port_conf, display_config_t::rows, port_config_t::rs485_rts_time_after_transmit, port_config_t::rs485_rts_time_before_transmit, display_config_t::scrollback, display_config_t::show_cursor, port_config_t::speed, port_config_t::stops, term_conf, and display_config_t::visual_bell.

Referenced by `check_configuration_file()`, and `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.1.4 dump_configuration_to_cli()

```
void dump_configuration_to_cli (
    char * section )
```

Print the serial port items

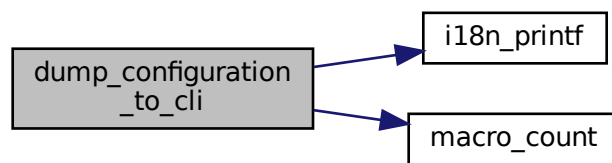
Print the terminal items

... and the macro's

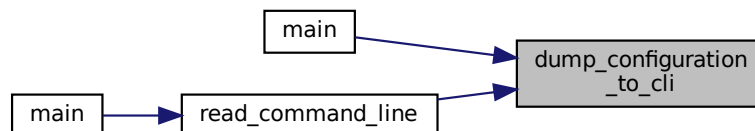
References `display_config_t::background_color`, `port_config_t::bits`, `display_config_t::block_cursor`, `display_config_t::char_queue`, `display_config_t::columns`, `display_config_t::crlfauto`, `display_config_t::delay`, `port_config_t::disable_port_lock`, `display_config_t::echo`, `port_config_t::flow_control`, `display_config_t::font`, `display_config_t::foreground_color`, `i18n_printf()`, `macro_count()`, `macros`, `port_config_t::parity`, `port_config_t::port`, `port_conf`, `display_config_t::rows`, `port_config_t::rs485_rts_time_after_transmit`, `port_config_t::rs485_rts_time_before_transmit`, `display_config_t::scrollback`, `display_config_t::show_cursor`, `port_config_t::speed`, `port_config_t::stops`, `term_conf`, `display_config_t::timestamp`, and `display_config_t::visual_bell`.

Referenced by `main()`, and `read_command_line()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.1.5 hard_default_configuration()

```
void hard_default_configuration (
    void )
```

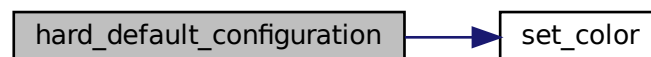
Create a new <default> configuration.

References `display_config_t::background_color`, `port_config_t::bits`, `display_config_t::block_cursor`, `display_config_t::char_queue`, `display_config_t::columns`, `display_config_t::crlfauto`, `DEFAULT_BITS`, `DEFAULT_CHAR`, `DEFAULT_DELAY`, `DEFAULT_DELAY_RS485`, `DEFAULT_ECHO`, `DEFAULT_FLOW`, `DEFAULT_FONT`, `DEFAULT_PARITY`, `DEFAULT_PORT`, `DEFAULT_SCROLLBACK`, `DEFAULT_SPEED`, `DEFAULT_STOP`, `display_config_t::delay`, `port_config_t::disable_port_lock`, `display_config_t::echo`,

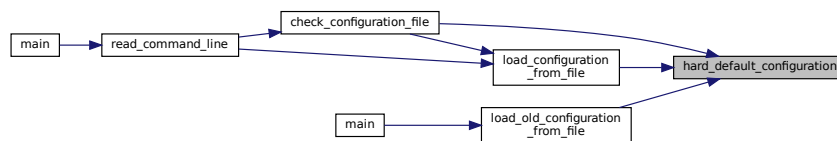
`port_config_t::flow_control`, `display_config_t::font`, `display_config_t::foreground_color`, `port_config_t::parity`, `port_config_t::port`, `port_conf`, `display_config_t::rows`, `port_config_t::rs485_rts_time_after_transmit`, `port_config_t::rs485_rts_time_before_transmit`, `display_config_t::scrollback`, `set_color()`, `display_config_t::show_cursor`, `port_config_t::speed`, `port_config_t::stops`, `term_conf`, `display_config_t::timestamp`, and `display_config_t::visual_bell`.

Referenced by `check_configuration_file()`, `load_configuration_from_file()`, and `load_old_configuration_from_file()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.1.6 load_configuration_from_file()

```
int load_configuration_from_file (
    const char * section )
```

Load the key file Note: all sections are loaded into memory.

Check if the <section> exists in the key file.

First initialize with a default structure. Not really needed.

The Font is a Pango structure. This only can be added to a terminal So we have to convert it.

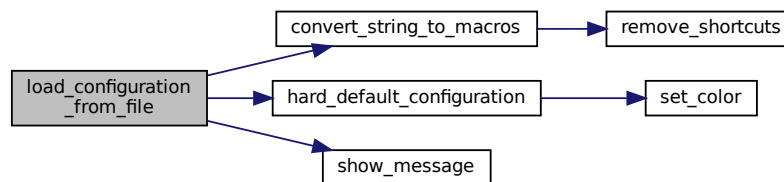
Convert the stringlist to macros. Existing shortcuts will be delete from `convert_string_to_macros`

References `display_config_t::background_color`, `port_config_t::bits`, `display_config_t::char_queue`, `display_config_t::columns`, `CONF_ITEM_BITS`, `CONF_ITEM_CRLF_AUTO`, `CONF_ITEM_DISABLE`, `_PORT_LOCK`, `CONF_ITEM_ECHO`, `CONF_ITEM_FLOW_CONTROL`, `CONF_ITEM_FONT`, `CONF_ITEM_MACROS`, `CONF_ITEM_PARITY`, `CONF_ITEM_PORT`, `CONF_ITEM_RS485_RTS_TIME_AFTER_TX`,

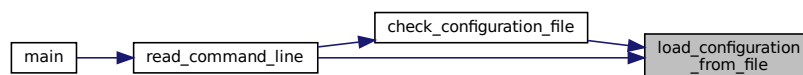
CONF_ITEM_RS485_RTS_TIME_BEFORE_TX, CONF_ITEM_SPEED, CONF_ITEM_STOPBITS, CONF_ITEM_TERM_BACKGROUND_ALPHA, CONF_ITEM_TERM_BACKGROUND_BLUE, CONF_ITEM_TERM_BACKGROUND_GREEN, CONF_ITEM_TERM_BACKGROUND_RED, CONF_ITEM_TERM_COLS, CONF_ITEM_TERM_FOREGROUND_ALPHA, CONF_ITEM_TERM_FOREGROUND_BLUE, CONF_ITEM_TERM_FOREGROUND_GREEN, CONF_ITEM_TERM_FOREGROUND_RED, CONF_ITEM_TERM_ROWS, CONF_ITEM_TERM_SCROLLBACK, CONF_ITEM_TERM_SHOW_CURSOR, CONF_ITEM_TERM_VISUAL_BELL, CONF_ITEM_WAIT_CHAR, CONF_ITEM_WAIT_DELAY, config_file, ConfigurationItem, convert_string_to_macros(), display_config_t::crlfauto, display_config_t::delay, port_config_t::disable_port_lock, display_config_t::echo, port_config_t::flow_control, display_config_t::font, display_config_t::foreground_color, hard_default_configuration(), MSG_ERR, port_config_t::parity, port_config_t::port, port_conf, display_config_t::rows, port_config_t::rs485_rts_time_after_transmit, port_config_t::rs485_rts_time_before_transmit, display_config_t::scrollback, display_config_t::show_cursor, show_message(), port_config_t::speed, port_config_t::stops, term_conf, and display_config_t::visual_bell.

Referenced by `check_configuration_file()`, and `read_command_line()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.1.7 remove_section()

```

int remove_section (
    char * cfg_file,
    char * section )

```

Remove a section from the file TODO: Perhaps remove because we dont need it.

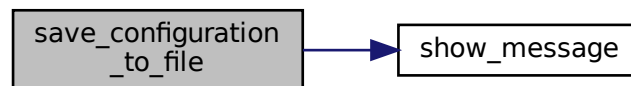
7.21.1.8 save_configuration_to_file()

```
void save_configuration_to_file (
    GKeyFile * config,
    const char * section )
```

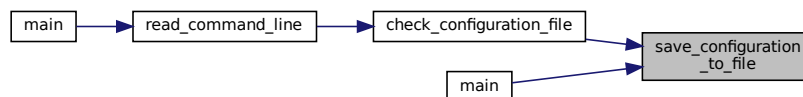
References **config**, **config_file**, **MSG_WRN**, and **show_message()**.

Referenced by **check_configuration_file()**, and **main()**.

Here is the call graph for this function:



Here is the caller graph for this function:



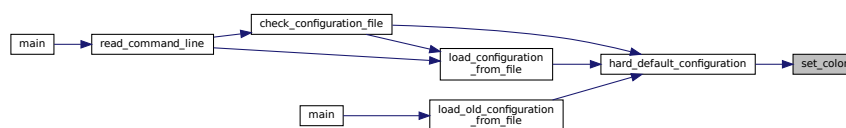
7.21.1.9 set_color()

```
void set_color (
    GdkRGBA * color,
    float R,
    float G,
    float B,
    float A )
```

Convert the colors RGB to internal color scheme.

Referenced by **hard_default_configuration()**.

Here is the caller graph for this function:



7.21.1.10 validate_configuration()

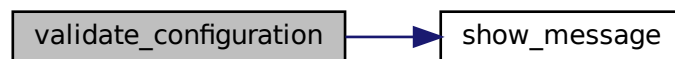
```
void validate_configuration (
    void )
```

validate the active configuration

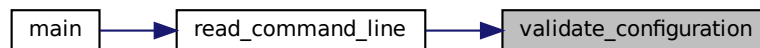
References `port_config_t::bits`, `DEFAULT_BITS`, `DEFAULT_DELAY`, `DEFAULT_FONT`, `DEFAULT_STOP`, `display_config_t::delay`, `display_config_t::font`, `MSG_ERR`, `port_conf`, `show_message()`, `port_config_t::speed`, `port_config_t::stops`, and `term_conf`.

Referenced by `read_command_line()`.

Here is the call graph for this function:



Here is the caller graph for this function:



7.21.2 Variable Documentation

7.21.2.1 config_file

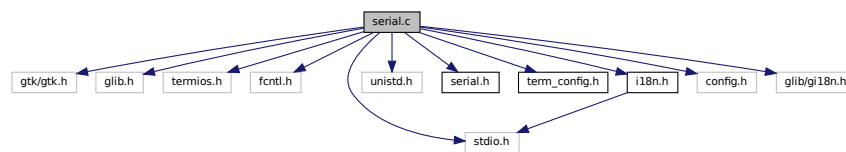
```
GFile* config_file [extern]
```

The key file.

Referenced by `check_configuration_file()`, `config_file_init()`, `load_configuration_from_file()`, `main()`, and `save_configuration_to_file()`.

7.22 serial.c File Reference

```
#include <gtk/gtk.h>
#include <glib.h>
#include <termios.h>
#include <fcntl.h>
#include <stdio.h>
#include <unistd.h>
#include "serial.h"
#include "term_config.h"
#include "i18n.h"
#include <config.h>
#include <glib/gi18n.h>
Include dependency graph for serial.c:
```



Functions

- `char * get_port_string (void)`

Variables

- `port_config_t port_conf`
- `struct termios termios_save`
- `int serial_port_fd = -1`

7.22.1 Function Documentation

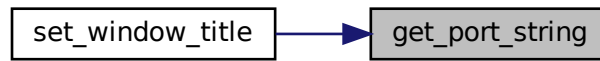
7.22.1.1 get_port_string()

```
char* get_port_string (
    void )
```

References `port_config_t::bits`, `parity`, `port_config_t::parity`, `port_config_t::port`, `port_conf`, `serial_port`, `_fd`, `port_config_t::speed`, and `port_config_t::stops`.

Referenced by `set_window_title()`.

Here is the caller graph for this function:



7.22.2 Variable Documentation

7.22.2.1 port_conf

```
port_config_t port_conf
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `get_port_string()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, `read_command_line()`, and `validate_configuration()`.

7.22.2.2 serial_port_fd

```
int serial_port_fd = -1
```

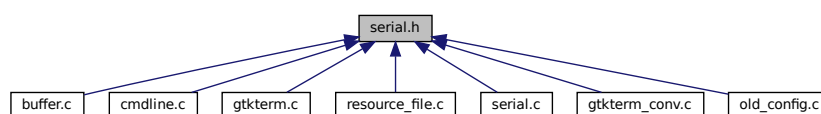
Referenced by `get_port_string()`.

7.22.2.3 termios_save

```
struct termios termios_save
```

7.23 serial.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- struct **port_config_t**

Macros

- **#define DEFAULT_PORT** "/dev/ttyS0"
- **#define DEFAULT_SPEED** 115200
- **#define DEFAULT_PARITY** 0
- **#define DEFAULT_BITS** 8
- **#define DEFAULT_STOP** 1
- **#define DEFAULT_FLOW** 0
- **#define RECEIVE_BUFFER** 8192
- **#define TRANSMIT_BUFFER** 4096
- **#define LINE_FEED** 0x0A
- **#define POLL_DELAY** 100 /* in ms (for control signals) */

Functions

- char * **get_port_string** (void)

Variables

- int **serial_port_fd**
- **port_config_t** **port_conf**

7.23.1 Macro Definition Documentation

7.23.1.1 DEFAULT_BITS

```
#define DEFAULT_BITS 8
```

7.23.1.2 DEFAULT_FLOW

```
#define DEFAULT_FLOW 0
```

7.23.1.3 DEFAULT_PARITY

```
#define DEFAULT_PARITY 0
```

7.23.1.4 DEFAULT_PORT

```
#define DEFAULT_PORT "/dev/ttyS0"
```

7.23.1.5 DEFAULT_SPEED

```
#define DEFAULT_SPEED 115200
```

7.23.1.6 DEFAULT_STOP

```
#define DEFAULT_STOP 1
```

7.23.1.7 LINE_FEED

```
#define LINE_FEED 0x0A
```

7.23.1.8 POLL_DELAY

```
#define POLL_DELAY 100 /* in ms (for control signals) */
```

7.23.1.9 RECEIVE_BUFFER

```
#define RECEIVE_BUFFER 8192
```

7.23.1.10 TRANSMIT_BUFFER

```
#define TRANSMIT_BUFFER 4096
```

7.23.2 Function Documentation

7.23.2.1 get_port_string()

```
char* get_port_string (
    void )
```

References `port_config_t::bits`, `parity`, `port_config_t::parity`, `port_config_t::port`, `port_conf`, `serial_port_↵
_fd`, `port_config_t::speed`, and `port_config_t::stops`.

Referenced by `set_window_title()`.

Here is the caller graph for this function:



7.23.3 Variable Documentation

7.23.3.1 port_conf

```
port_config_t port_conf [extern]
```

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `get_port_string()`, `hard_default_↵
configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, `read_command_↵
line()`, and `validate_configuration()`.

7.23.3.2 serial_port_fd

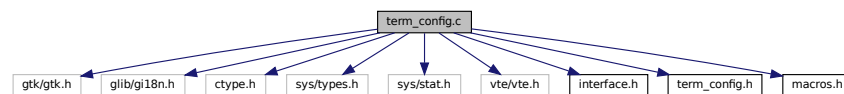
```
int serial_port_fd [extern]
```

Referenced by `get_port_string()`.

7.24 term_config.c File Reference

```
#include <gtk/gtk.h>
#include <glib/glib.h>
#include <ctype.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <vte/vte.h>
#include "interface.h"
#include "term_config.h"
#include "macros.h"
```

Include dependency graph for term_config.c:



Macros

- #define **CONFIGURATION_FILENAME** ".gtktermrc"

Variables

- **display_config_t** term_conf

7.24.1 Macro Definition Documentation

7.24.1.1 CONFIGURATION_FILENAME

```
#define CONFIGURATION_FILENAME ".gtktermrc"
```

7.24.2 Variable Documentation

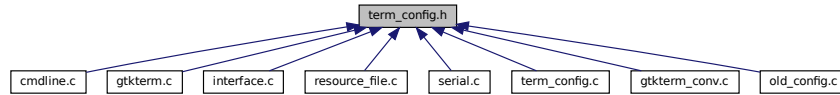
7.24.2.1 term_conf

```
display_config_t term_conf
```

Referenced by **copy_configuration()**, **dump_configuration_to_cli()**, **hard_default_configuration()**, **load_configuration_from_file()**, **load_old_configuration_from_file()**, **read_command_line()**, and **validate_configuration()**.

7.25 term_config.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- struct **display_config_t**

Macros

- #define **DEFAULT_FONT** "Monospace 12"
- #define **DEFAULT_SCROLLBACK** 10000
- #define **DEFAULT_DELAY** 0
- #define **DEFAULT_CHAR** -1
- #define **DEFAULT_DELAY_RS485** 30
- #define **DEFAULT_ECHO** FALSE

Variables

- **display_config_t** **term_conf**

Define external variables here configuration for terminal window and serial port.

7.25.1 Macro Definition Documentation

7.25.1.1 DEFAULT_CHAR

```
#define DEFAULT_CHAR -1
```

7.25.1.2 DEFAULT_DELAY

```
#define DEFAULT_DELAY 0
```

7.25.1.3 DEFAULT_DELAY_RS485

```
#define DEFAULT_DELAY_RS485 30
```

7.25.1.4 DEFAULT_ECHO

```
#define DEFAULT_ECHO FALSE
```

7.25.1.5 DEFAULT_FONT

```
#define DEFAULT_FONT "Monospace 12"
```

7.25.1.6 DEFAULT_SCROLLBACK

```
#define DEFAULT_SCROLLBACK 10000
```

7.25.2 Variable Documentation

7.25.2.1 term_conf

```
display_config_t term_conf [extern]
```

Define external variables here configuration for terminal window and serial port.

Referenced by `copy_configuration()`, `dump_configuration_to_cli()`, `hard_default_configuration()`, `load_configuration_from_file()`, `load_old_configuration_from_file()`, `read_command_line()`, and `validate_configuration()`.

Index

- action
 - macro_t, 22
- add_input
 - files.h, 38
- add_shortcuts
 - macros.h, 58
- ASCII_VIEW
 - interface.h, 52
- background_alpha
 - old_config.c, 63
- background_blue
 - old_config.c, 63
- background_color
 - display_config_t, 16
- background_green
 - old_config.c, 63
- background_red
 - old_config.c, 64
- bits
 - old_config.c, 64
 - port_config_t, 23
- block_cursor
 - display_config_t, 16
 - old_config.c, 64
- buffer
 - GtkTermWindow, 20
- buffer.c, 27
 - clear_buffer, 28
 - clear_func, 30
 - create_buffer, 28
 - delete_buffer, 28
 - insert_timestamp, 28
 - overlapped, 30
 - put_chars, 29
 - set_clear_func, 29
 - set_display_func, 29
 - timestamp_on, 31
 - TIMESTAMP_SIZE, 28
 - unset_clear_func, 29
 - unset_display_func, 29
 - virt_col_pos, 31
 - write_buffer, 29
 - write_buffer_with_func, 30
 - write_func, 31
- buffer.h, 31
 - BUFFER_SIZE, 32
 - clear_buffer, 32
 - create_buffer, 32
 - delete_buffer, 32
 - put_chars, 33
 - set_clear_func, 33
 - set_display_func, 33
 - unset_clear_func, 33
 - unset_display_func, 33
 - write_buffer, 33
 - write_buffer_with_func, 34
- BUFFER_SIZE
 - buffer.h, 32
- cfg
 - gtkterm_conv.c, 45
 - old_config.c, 64
- CFG_BOOL
 - parsecfg.h, 74
- CFG_BOOL_ERROR
 - parsecfg.h, 73
- CFG_CREATE_FAIL
 - parsecfg.h, 73
- CFG_DOUBLE
 - parsecfg.h, 74
- CFG_DOUBLE_QUOTE
 - parsecfg.h, 74
- CFG_END
 - parsecfg.h, 74
- CFG_FLOAT
 - parsecfg.h, 74
- CFG_INI
 - parsecfg.h, 73
- CFG_INT
 - parsecfg.h, 74
- CFG_INTERNAL_ERROR
 - parsecfg.h, 73
- CFG_INVALID_NUMBER
 - parsecfg.h, 73
- CFG_JUST_RETURN_WITHOUT_MSG
 - parsecfg.h, 73
- CFG_LONG
 - parsecfg.h, 74
- CFG_MEM_ALLOC_FAIL
 - parsecfg.h, 73
- CFG_NO_CLOSING_BRACE
 - parsecfg.h, 73
- CFG_NO_ERROR
 - parsecfg.h, 73
- CFG_NO_QUOTE
 - parsecfg.h, 74
- CFG_OPEN_FAIL
 - parsecfg.h, 73
- CFG_OUT_OF_RANGE

- parsecfg.h, 73
- CFG_PARAMETER
 - parsecfg.h, 73
- CFG_SECTION
 - parsecfg.h, 73
- CFG_SIMPLE
 - parsecfg.h, 73
- CFG_SINGLE_QUOTE
 - parsecfg.h, 74
- CFG_STRING
 - parsecfg.h, 74
- CFG_STRING_LIST
 - parsecfg.h, 74
- CFG_SYNTAX_ERROR
 - parsecfg.h, 73
- CFG_UINT
 - parsecfg.h, 74
- CFG_ULONG
 - parsecfg.h, 74
- CFG_USED_SECTION
 - parsecfg.h, 73
- CFG_VALUE
 - parsecfg.h, 73
- CFG_WRONG_PARAMETER
 - parsecfg.h, 73
- cfgAllocForNewSection
 - parsecfg.c, 69
 - parsecfg.h, 74
- cfgDump
 - parsecfg.c, 69
 - parsecfg.h, 74
- cfgErrorCode
 - parsecfg.h, 72
- cfgFileType
 - parsecfg.h, 73
- cfgKeywordValue
 - parsecfg.h, 73
- cfgList
 - parsecfg.h, 72
- cfgList_tag, 13
 - next, 13
 - str, 13
- cfgParse
 - parsecfg.c, 69
 - parsecfg.h, 74
- cfgQuote
 - parsecfg.h, 73
- cfgSectionNameToNumber
 - parsecfg.c, 69
 - parsecfg.h, 75
- cfgSectionNumberToName
 - parsecfg.c, 70
 - parsecfg.h, 75
- cfgSetFatalFunc
 - parsecfg.c, 70
 - parsecfg.h, 75
- cfgStoreValue
 - parsecfg.c, 70
- parsecfg.h, 76
- cfgStruct, 14
 - parameterName, 14
 - type, 14
 - value, 15
- cfgValueType
 - parsecfg.h, 74
- char_queue
 - display_config_t, 16
 - port_config_t, 23
- check_configuration_file
 - resource_file.c, 79
 - resource_file.h, 88
- clear_buffer
 - buffer.c, 28
 - buffer.h, 32
- clear_func
 - buffer.c, 30
- closure
 - macro_t, 22
- cmdline.c, 34
 - config, 36
 - display_help, 35
 - read_command_line, 35
- cmdline.h, 37
 - read_command_line, 37
- COLUMN_ACTION
 - macros.c, 54
- COLUMN_SHORTCUT
 - macros.c, 54
- columns
 - display_config_t, 16
 - old_config.c, 64
- CONF_ITEM_BITS
 - resource_file.c, 78
- CONF_ITEM_CRLF_AUTO
 - resource_file.c, 78
- CONF_ITEM_DISABLE_PORT_LOCK
 - resource_file.c, 78
- CONF_ITEM_ECHO
 - resource_file.c, 78
- CONF_ITEM_FLOW_CONTROL
 - resource_file.c, 78
- CONF_ITEM_FONT
 - resource_file.c, 78
- CONF_ITEM_MACROS
 - resource_file.c, 78
- CONF_ITEM_PARITY
 - resource_file.c, 78
- CONF_ITEM_PORT
 - resource_file.c, 78
- CONF_ITEM_RS485_RTS_TIME_AFTER_TX
 - resource_file.c, 78
- CONF_ITEM_RS485_RTS_TIME_BEFORE_TX
 - resource_file.c, 78
- CONF_ITEM_SPEED
 - resource_file.c, 78
- CONF_ITEM_STOPBITS

- resource_file.c, 78
- CONF_ITEM_TERM_BACKGROUND_ALPHA
 - resource_file.c, 78
- CONF_ITEM_TERM_BACKGROUND_BLUE
 - resource_file.c, 78
- CONF_ITEM_TERM_BACKGROUND_GREEN
 - resource_file.c, 78
- CONF_ITEM_TERM_BACKGROUND_RED
 - resource_file.c, 78
- CONF_ITEM_TERM_COLS
 - resource_file.c, 78
- CONF_ITEM_TERM_FOREGROUND_ALPHA
 - resource_file.c, 78
- CONF_ITEM_TERM_FOREGROUND_BLUE
 - resource_file.c, 78
- CONF_ITEM_TERM_FOREGROUND_GREEN
 - resource_file.c, 78
- CONF_ITEM_TERM_FOREGROUND_RED
 - resource_file.c, 78
- CONF_ITEM_TERM_ROWS
 - resource_file.c, 78
- CONF_ITEM_TERM_SCROLLBACK
 - resource_file.c, 78
- CONF_ITEM_TERM_SHOW_CURSOR
 - resource_file.c, 78
- CONF_ITEM_TERM_VISUAL_BELL
 - resource_file.c, 78
- CONF_ITEM_WAIT_CHAR
 - resource_file.c, 78
- CONF_ITEM_WAIT_DELAY
 - resource_file.c, 78
- config
 - cmdline.c, 36
 - interface.c, 50
- config_file
 - resource_file.c, 86
 - resource_file.h, 95
- config_file_init
 - resource_file.c, 79
 - resource_file.h, 89
- CONFIGURATION_FILENAME
 - resource_file.c, 77
 - term_config.c, 101
- ConfigurationItem
 - resource_file.c, 86
- convert_macros_to_string
 - macros.c, 55
 - macros.h, 58
- convert_string_to_macros
 - macros.c, 55
 - macros.h, 58
- copy_configuration
 - resource_file.c, 80
 - resource_file.h, 89
- create_buffer
 - buffer.c, 28
 - buffer.h, 32
- create_shortcuts
 - old_config.c, 62
- crlfauto
 - display_config_t, 17
 - old_config.c, 64
- DEFAULT_BITS
 - serial.h, 98
- DEFAULT_CHAR
 - term_config.h, 102
- DEFAULT_DELAY
 - term_config.h, 102
- DEFAULT_DELAY_RS485
 - term_config.h, 102
- DEFAULT_ECHO
 - term_config.h, 103
- default_filename
 - files.c, 38
 - files.h, 39
- DEFAULT_FLOW
 - serial.h, 98
- DEFAULT_FONT
 - term_config.h, 103
- DEFAULT_PARITY
 - serial.h, 98
- DEFAULT_PORT
 - serial.h, 98
- DEFAULT_SCROLLBACK
 - term_config.h, 103
- DEFAULT_SPEED
 - serial.h, 99
- DEFAULT_STOP
 - serial.h, 99
- delay
 - display_config_t, 17
- delete_buffer
 - buffer.c, 28
 - buffer.h, 32
- disable_port_lock
 - port_config_t, 24
- display
 - interface.c, 50
 - interface.h, 53
- display_config_t, 15
 - background_color, 16
 - block_cursor, 16
 - char_queue, 16
 - columns, 16
 - crlfauto, 17
 - delay, 17
 - echo, 17
 - font, 17
 - foreground_color, 17
 - rows, 18
 - scrollback, 18
 - show_cursor, 18
 - timestamp, 18
 - visual_bell, 18
- display_help
 - cmdline.c, 35

- dump_configuration_to_cli
 - resource_file.c, 81
 - resource_file.h, 90
- echo
 - display_config_t, 17
 - old_config.c, 65
- fetchVarFromCfgFile
 - parsecfg.c, 70
 - parsecfg.h, 76
- files.c, 37
 - default_filename, 38
- files.h, 38
 - add_input, 38
 - default_filename, 39
 - save_raw_file, 39
 - send_raw_file, 39
 - waiting_for_char, 39
- flow
 - old_config.c, 65
- flow_control
 - port_config_t, 24
- font
 - display_config_t, 17
 - old_config.c, 65
- foreground_alpha
 - old_config.c, 65
- foreground_blue
 - old_config.c, 65
- foreground_color
 - display_config_t, 17
- foreground_green
 - old_config.c, 65
- foreground_red
 - old_config.c, 66
- fullscreen
 - GtkTermWindow, 20
- get_port_string
 - serial.c, 96
 - serial.h, 99
- get_shortcuts
 - macros.c, 56
 - macros.h, 59
- GtkTerm
 - gtkterm.c, 40
- gtkterm.c, 40
 - GtkTerm, 40
 - GtkTermClass, 40
 - GtkTermWindowClass, 41
 - main, 41
 - set_window_title, 41
- gtkterm_conv.c, 42
 - cfg, 45
 - load_old_configuration_from_file, 42
 - main, 43
 - port_conf, 45
 - show_message, 44
 - term_conf, 45
- GtkTermClass
 - gtkterm.c, 40
- GtkTermWindow, 19
 - buffer, 20
 - fullscreen, 20
 - height, 20
 - infobar, 20
 - maximized, 20
 - menubutton, 20
 - message, 20
 - parent_instance, 20
 - status, 21
 - toolmenu, 21
 - width, 21
- GtkTermWindowClass
 - gtkterm.c, 41
- hard_default_configuration
 - resource_file.c, 82
 - resource_file.h, 91
- height
 - GtkTermWindow, 20
- HEXADECIMAL_VIEW
 - interface.h, 52
- i18n.c, 46
 - i18n_fprintf, 46
 - i18n_perror, 46
 - i18n_printf, 46
 - strerror_utf8, 47
- i18n.h, 47
 - i18n_fprintf, 48
 - I18N_H, 48
 - i18n_perror, 48
 - i18n_printf, 48
 - strerror_utf8, 49
- i18n_fprintf
 - i18n.c, 46
 - i18n.h, 48
- I18N_H
 - i18n.h, 48
- i18n_perror
 - i18n.c, 46
 - i18n.h, 48
- i18n_printf
 - i18n.c, 46
 - i18n.h, 48
- infobar
 - GtkTermWindow, 20
- insert_timestamp
 - buffer.c, 28
- install_prefix
 - meson_post_install, 11
- interface.c, 49
 - config, 50
 - display, 50
 - show_message, 50
 - timestamp_on, 51

- virt_col_pos, 51
- interface.h, 51
 - ASCII_VIEW, 52
 - display, 53
 - HEXADECIMAL_VIEW, 52
 - MSG_ERR, 52
 - MSG_WRN, 52
 - show_message, 52
 - Text, 53
- LINE_FEED
 - serial.h, 99
- load_configuration_from_file
 - resource_file.c, 83
 - resource_file.h, 92
- load_old_configuration_from_file
 - gtkterm_conv.c, 42
 - old_config.c, 62
- macro_count
 - macros.c, 56
 - macros.h, 59
- macro_list
 - old_config.c, 66
- macro_t, 21
 - action, 22
 - closure, 22
 - shortcut, 22
- macros
 - macros.c, 57
 - macros.h, 60
- macros.c, 53
 - COLUMN_ACTION, 54
 - COLUMN_SHORTCUT, 54
 - convert_macros_to_string, 55
 - convert_string_to_macros, 55
 - get_shortcuts, 56
 - macro_count, 56
 - macros, 57
 - nr_of_macros, 57
 - NUM_COLUMNS, 54
 - remove_shortcuts, 56
- macros.h, 57
 - add_shortcuts, 58
 - convert_macros_to_string, 58
 - convert_string_to_macros, 58
 - get_shortcuts, 59
 - macro_count, 59
 - macros, 60
 - remove_shortcuts, 59
- main
 - gtkterm.c, 41
 - gtkterm_conv.c, 43
- maximized
 - GtkTermWindow, 20
- menubutton
 - GtkTermWindow, 20
- meson_post_install, 11
 - install_prefix, 11
- schemadir, 11
- meson_post_install.py, 60
- message
 - GtkTermWindow, 20
- MSG_ERR
 - interface.h, 52
- MSG_WRN
 - interface.h, 52
- next
 - cfgList_tag, 13
- nr_of_macros
 - macros.c, 57
 - old_config.c, 66
- NUM_COLUMNS
 - macros.c, 54
- old_config.c, 61
 - background_alpha, 63
 - background_blue, 63
 - background_green, 63
 - background_red, 64
 - bits, 64
 - block_cursor, 64
 - cfg, 64
 - columns, 64
 - create_shortcuts, 62
 - crlfauto, 64
 - echo, 65
 - flow, 65
 - font, 65
 - foreground_alpha, 65
 - foreground_blue, 65
 - foreground_green, 65
 - foreground_red, 66
 - load_old_configuration_from_file, 62
 - macro_list, 66
 - nr_of_macros, 66
 - parity, 66
 - port, 66
 - rows, 66
 - rts_time_after_tx, 67
 - rts_time_before_tx, 67
 - scrollback, 67
 - speed, 67
 - stopbits, 67
 - timestamp, 67
 - visual_bell, 68
 - wait_char, 68
 - wait_delay, 68
- overlapped
 - buffer.c, 30
- parameterName
 - cfgStruct, 14
- parent_instance
 - GtkTermWindow, 20
- parity
 - old_config.c, 66

- port_config_t, 24
- parsecfg.c, 68
 - cfgAllocForNewSection, 69
 - cfgDump, 69
 - cfgParse, 69
 - cfgSectionNameToNumber, 69
 - cfgSectionNumberToName, 70
 - cfgSetFatalFunc, 70
 - cfgStoreValue, 70
 - fetchVarFromCfgFile, 70
- parsecfg.h, 71
 - CFG_BOOL, 74
 - CFG_BOOL_ERROR, 73
 - CFG_CREATE_FAIL, 73
 - CFG_DOUBLE, 74
 - CFG_DOUBLE_QUOTE, 74
 - CFG_END, 74
 - CFG_FLOAT, 74
 - CFG_INI, 73
 - CFG_INT, 74
 - CFG_INTERNAL_ERROR, 73
 - CFG_INVALID_NUMBER, 73
 - CFG_JUST_RETURN_WITHOUT_MSG, 73
 - CFG_LONG, 74
 - CFG_MEM_ALLOC_FAIL, 73
 - CFG_NO_CLOSING_BRACE, 73
 - CFG_NO_ERROR, 73
 - CFG_NO_QUOTE, 74
 - CFG_OPEN_FAIL, 73
 - CFG_OUT_OF_RANGE, 73
 - CFG_PARAMETER, 73
 - CFG_SECTION, 73
 - CFG_SIMPLE, 73
 - CFG_SINGLE_QUOTE, 74
 - CFG_STRING, 74
 - CFG_STRING_LIST, 74
 - CFG_SYNTAX_ERROR, 73
 - CFG_UINT, 74
 - CFG_ULONG, 74
 - CFG_USED_SECTION, 73
 - CFG_VALUE, 73
 - CFG_WRONG_PARAMETER, 73
 - cfgAllocForNewSection, 74
 - cfgDump, 74
 - cfgErrorCode, 72
 - cfgFileType, 73
 - cfgKeywordValue, 73
 - cfgList, 72
 - cfgParse, 74
 - cfgQuote, 73
 - cfgSectionNameToNumber, 75
 - cfgSectionNumberToName, 75
 - cfgSetFatalFunc, 75
 - cfgStoreValue, 76
 - cfgValueType, 74
 - fetchVarFromCfgFile, 76
 - PARSECFG_VERSION, 72
- PARSECFG_VERSION
- parsecfg.h, 72
- POLL_DELAY
 - serial.h, 99
- port
 - old_config.c, 66
 - port_config_t, 24
- port_conf
 - gtkterm_conv.c, 45
 - serial.c, 97
 - serial.h, 100
- port_config_t, 23
 - bits, 23
 - char_queue, 23
 - disable_port_lock, 24
 - flow_control, 24
 - parity, 24
 - port, 24
 - rs485_rts_time_after_transmit, 24
 - rs485_rts_time_before_transmit, 24
 - speed, 25
 - stops, 25
- put_chars
 - buffer.c, 29
 - buffer.h, 33
- read_command_line
 - cmdline.c, 35
 - cmdline.h, 37
- README.md, 76
- RECEIVE_BUFFER
 - serial.h, 99
- remove_section
 - resource_file.c, 84
 - resource_file.h, 93
- remove_shortcuts
 - macros.c, 56
 - macros.h, 59
- resource_file.c, 76
 - check_configuration_file, 79
 - CONF_ITEM_BITS, 78
 - CONF_ITEM_CRLF_AUTO, 78
 - CONF_ITEM_DISABLE_PORT_LOCK, 78
 - CONF_ITEM_ECHO, 78
 - CONF_ITEM_FLOW_CONTROL, 78
 - CONF_ITEM_FONT, 78
 - CONF_ITEM_MACROS, 78
 - CONF_ITEM_PARITY, 78
 - CONF_ITEM_PORT, 78
 - CONF_ITEM_RS485_RTS_TIME_AFTER_TX, 78
 - CONF_ITEM_RS485_RTS_TIME_BEFORE_TX, 78
 - CONF_ITEM_SPEED, 78
 - CONF_ITEM_STOPBITS, 78
 - CONF_ITEM_TERM_BACKGROUND_ALPHA, 78
 - CONF_ITEM_TERM_BACKGROUND_BLUE, 78
 - CONF_ITEM_TERM_BACKGROUND_GREEN, 78
 - CONF_ITEM_TERM_BACKGROUND_RED, 78
 - CONF_ITEM_TERM_COLS, 78

- CONF_ITEM_TERM_FOREGROUND_ALPHA, 78
- CONF_ITEM_TERM_FOREGROUND_BLUE, 78
- CONF_ITEM_TERM_FOREGROUND_GREEN, 78
- CONF_ITEM_TERM_FOREGROUND_RED, 78
- CONF_ITEM_TERM_ROWS, 78
- CONF_ITEM_TERM_SCROLLBACK, 78
- CONF_ITEM_TERM_SHOW_CURSOR, 78
- CONF_ITEM_TERM_VISUAL_BELL, 78
- CONF_ITEM_WAIT_CHAR, 78
- CONF_ITEM_WAIT_DELAY, 78
- config_file, 86
- config_file_init, 79
- CONFIGURATION_FILENAME, 77
- ConfigurationItem, 86
- copy_configuration, 80
- dump_configuration_to_cli, 81
- hard_default_configuration, 82
- load_configuration_from_file, 83
- remove_section, 84
- save_configuration_to_file, 84
- set_color, 85
- validate_configuration, 85
- resource_file.h, 87
 - check_configuration_file, 88
 - config_file, 95
 - config_file_init, 89
 - copy_configuration, 89
 - dump_configuration_to_cli, 90
 - hard_default_configuration, 91
 - load_configuration_from_file, 92
 - remove_section, 93
 - save_configuration_to_file, 93
 - set_color, 94
 - validate_configuration, 94
- rows
 - display_config_t, 18
 - old_config.c, 66
- rs485_rts_time_after_transmit
 - port_config_t, 24
- rs485_rts_time_before_transmit
 - port_config_t, 24
- rts_time_after_tx
 - old_config.c, 67
- rts_time_before_tx
 - old_config.c, 67
- save_configuration_to_file
 - resource_file.c, 84
 - resource_file.h, 93
- save_raw_file
 - files.h, 39
- schemadir
 - meson_post_install, 11
- scrollback
 - display_config_t, 18
 - old_config.c, 67
- send_raw_file
 - files.h, 39
- serial.c, 96
 - get_port_string, 96
 - port_conf, 97
 - serial_port_fd, 97
 - termios_save, 97
- serial.h, 97
 - DEFAULT_BITS, 98
 - DEFAULT_FLOW, 98
 - DEFAULT_PARITY, 98
 - DEFAULT_PORT, 98
 - DEFAULT_SPEED, 99
 - DEFAULT_STOP, 99
 - get_port_string, 99
 - LINE_FEED, 99
 - POLL_DELAY, 99
 - port_conf, 100
 - RECEIVE_BUFFER, 99
 - serial_port_fd, 100
 - TRANSMIT_BUFFER, 99
- serial_port_fd
 - serial.c, 97
 - serial.h, 100
- set_clear_func
 - buffer.c, 29
 - buffer.h, 33
- set_color
 - resource_file.c, 85
 - resource_file.h, 94
- set_display_func
 - buffer.c, 29
 - buffer.h, 33
- set_window_title
 - gtkterm.c, 41
- shortcut
 - macro_t, 22
- show_cursor
 - display_config_t, 18
- show_message
 - gtkterm_conv.c, 44
 - interface.c, 50
 - interface.h, 52
- speed
 - old_config.c, 67
 - port_config_t, 25
- status
 - GtkTermWindow, 21
- stopbits
 - old_config.c, 67
- stops
 - port_config_t, 25
- str
 - cfgList_tag, 13
- strerror_utf8
 - i18n.c, 47
 - i18n.h, 49
- term_conf
 - gtkterm_conv.c, 45
 - term_config.c, 101

- term_config.h, 103
- term_config.c, 101
 - CONFIGURATION_FILENAME, 101
 - term_conf, 101
- term_config.h, 102
 - DEFAULT_CHAR, 102
 - DEFAULT_DELAY, 102
 - DEFAULT_DELAY_RS485, 102
 - DEFAULT_ECHO, 103
 - DEFAULT_FONT, 103
 - DEFAULT_SCROLLBACK, 103
 - term_conf, 103
- termios_save
 - serial.c, 97
- Text
 - interface.h, 53
- timestamp
 - display_config_t, 18
 - old_config.c, 67
- timestamp_on
 - buffer.c, 31
 - interface.c, 51
- TIMESTAMP_SIZE
 - buffer.c, 28
- toolmenu
 - GtkTermWindow, 21
- TRANSMIT_BUFFER
 - serial.h, 99
- type
 - cfgStruct, 14
- unset_clear_func
 - buffer.c, 29
 - buffer.h, 33
- unset_display_func
 - buffer.c, 29
 - buffer.h, 33
- validate_configuration
 - resource_file.c, 85
 - resource_file.h, 94
- value
 - cfgStruct, 15
- virt_col_pos
 - buffer.c, 31
 - interface.c, 51
- visual_bell
 - display_config_t, 18
 - old_config.c, 68
- wait_char
 - old_config.c, 68
- wait_delay
 - old_config.c, 68
- waiting_for_char
 - files.h, 39
- width
 - GtkTermWindow, 21
- write_buffer
 - buffer.c, 29
 - buffer.h, 33
 - write_buffer_with_func
 - buffer.c, 30
 - buffer.h, 34
 - write_func
 - buffer.c, 31