gtkterm

1.99.0

Generated by Doxygen 1.9.1

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
`<ctrl><shift>L`	Clear screen
`<ctrl><shift>R`	Send file
`<ctrl><shift>Q`	Quit
`<ctrl><shift>S`	Configure port
`<ctrl><shift>V`	Paste
`<ctrl><shift>C`	Copy
`<ctrl><shift>F`	Find
`<ctrl><shift>K`	Clear Scrollback
`<ctrl><shift>A`	Select All
`<ctrl><shift>B`	Send Break
`<ctrl>B`	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	killall -USR1 gtkterm
SIGUSR2	Close Port	killall -USR2 gtkterm

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

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

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')
```

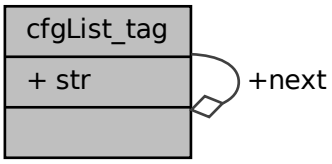
Chapter 6

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()**.

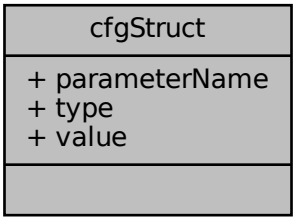
The documentation for this struct was generated from the following file:

- **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 scrollback

```
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
```

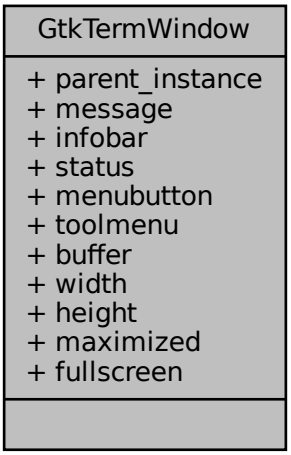
Referenced by **copy_configuration()**, **dump_configuration_to_cli()**, **hard_default_configuration()**, **load_configuration_from_file()**, and **load_old_configuration_from_file()**.

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:

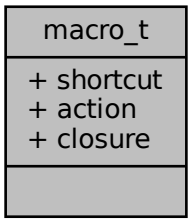
- **gtkterm.c**

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()**.

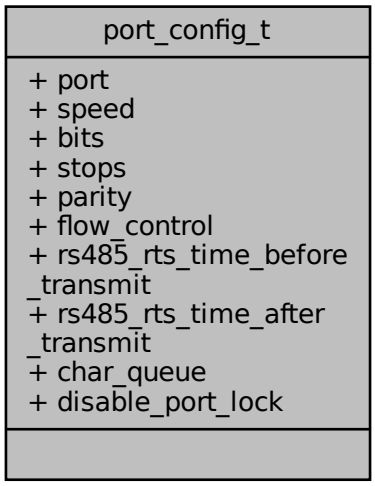
The documentation for this struct was generated from the following file:

- **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**

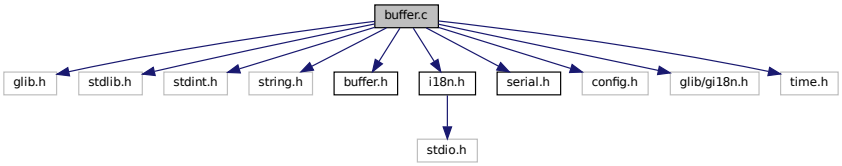
Chapter 7

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**.

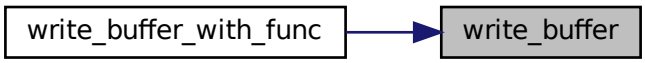
7.1.2.10 write_buffer()

```
void write_buffer (
            void  )
```

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

Referenced by **write_buffer_with_func()**.

Here is the caller graph for this function:

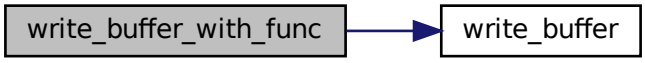


7.1.2.11 write_buffer_with_func()

```
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]
```

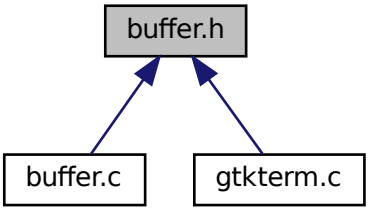
7.1.3.5 write_func

```
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 )
```

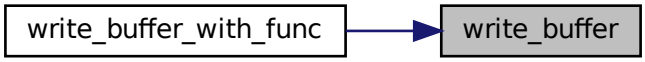
7.2.2.9 write_buffer()

```
void write_buffer (
            void  )
```

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

Referenced by **write_buffer_with_func()**.

Here is the caller graph for this function:



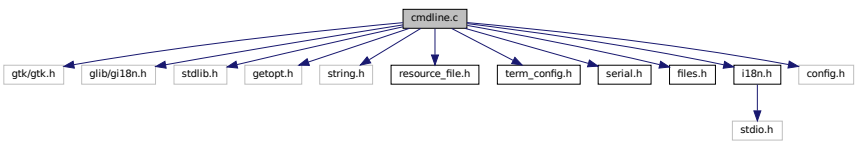
7.2.2.10 write_buffer_with_func()

```
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()

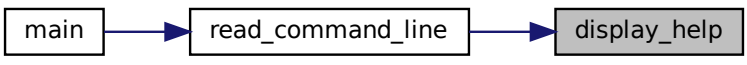
```
void display_help (
            void  )
```

References **i18n_printf()**.

Referenced by **read_command_line()**.

Here is the call graph for this function:

Here is the caller graph for this function:



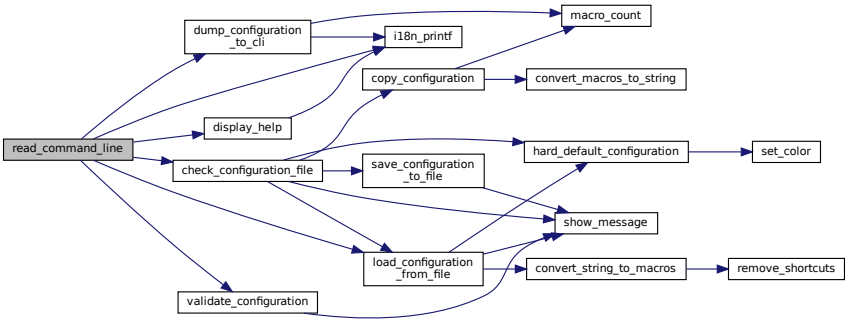
7.3.1.2 read_command_line()

```
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

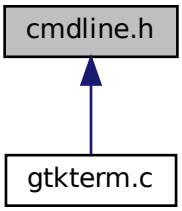
7.3.2.1 config

```
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

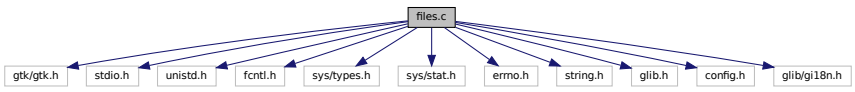
7.4.1.1 read_command_line()

```
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/gi18n.h>
```

Include dependency graph for files.c:



Variables

- char * **default_filename** = NULL

7.5.1 Variable Documentation

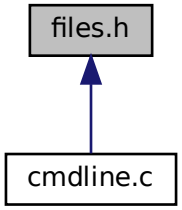
7.5.1.1 default_filename

```
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()**.

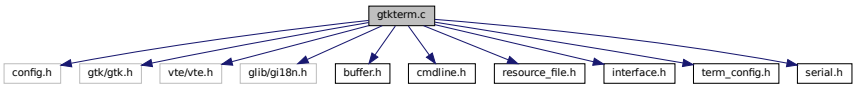
7.6.2.2 waiting_for_char

```
gboolean waiting_for_char  [extern]
```

7.7 gtkterm.c File Reference

```
#include "config.h"
#include <gtk/gtk.h>
#include <vte/vte.h>
#include <glib/gi18n.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
```

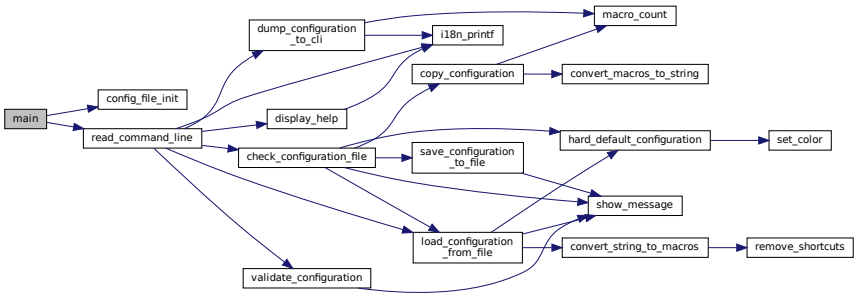
7.7.2 Function Documentation

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 )
```

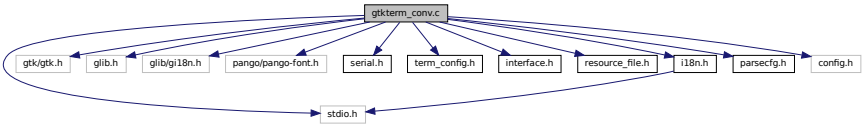
References **get_port_string()**.

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()

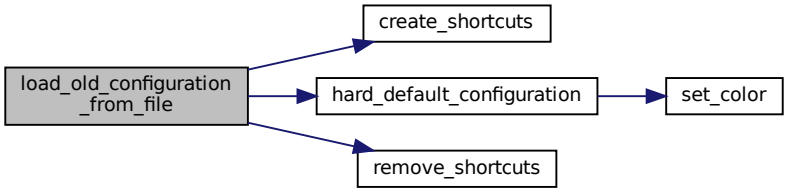
```
int load_old_configuration_from_file (
            int section_nr )
```

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

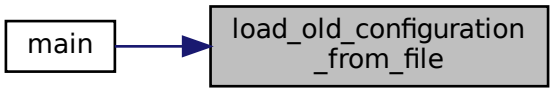
References **macro_t::action**, **background_alpha**, **background_blue**, **display_config_t::background_color**, **background_green**, **background_red**, **bits**, **port_config_t::bits**, **block_cursor**, **display_config_t::block_cursor**, **display_config_t::char_queue**, **columns**, **display_config_t::columns**, **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_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:



7.8.1.2 main()

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

Initialize for localization

Check if the file exists

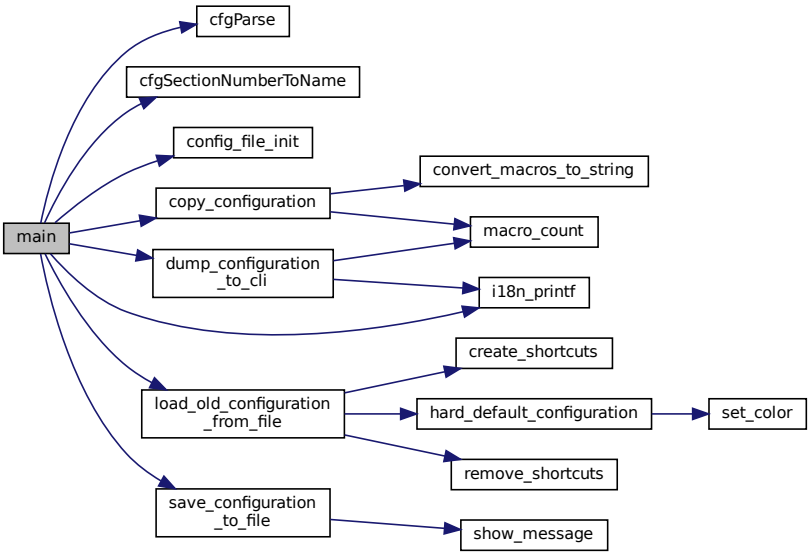
load old config file with parsecfg

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:



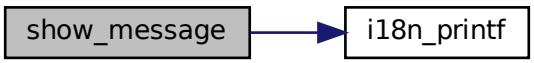
7.8.1.3 show_message()

```
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

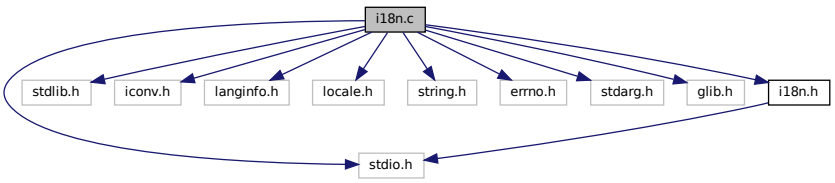
```
display_config_t term_conf
```

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()

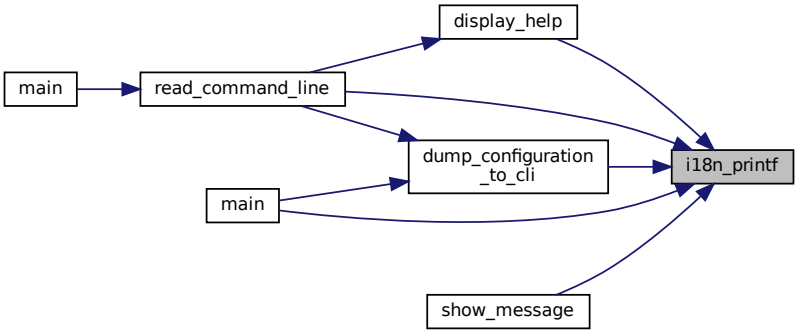
```
void i18n_perror (
            const char * s )
```

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:



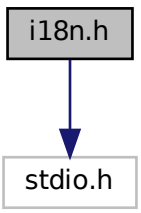
7.9.1.4 strerror_utf8()

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

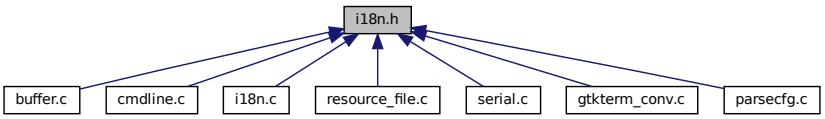
7.10 i18n.h File Reference

```
#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()

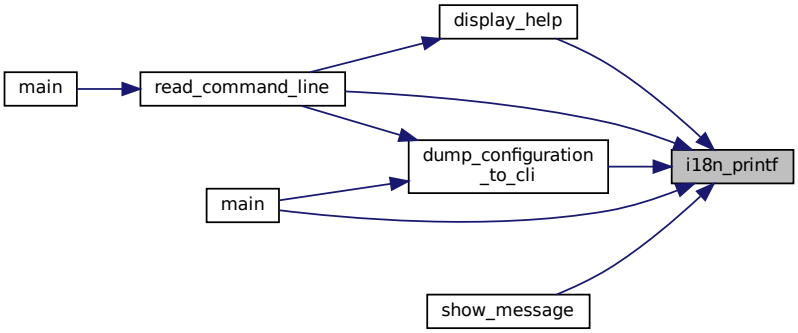
```
void i18n_perror (
            const char * s )
```

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:



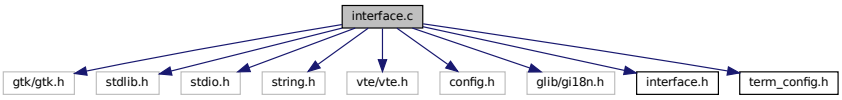
7.10.2.4 strerror_utf8()

```
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

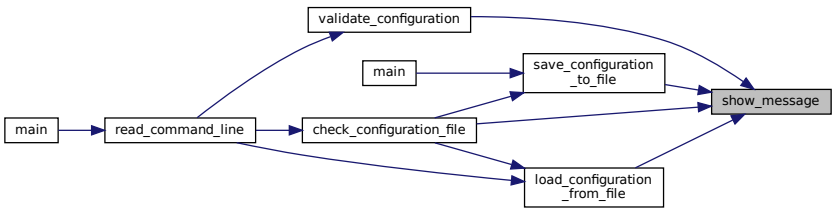
7.11.1 Function Documentation

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
```

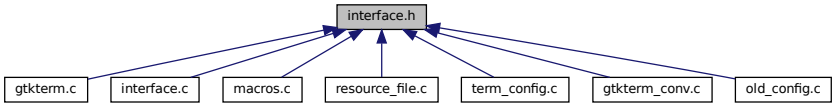
Referenced by **put_chars()**.

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()

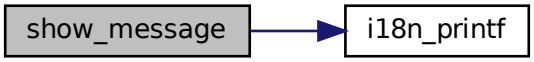
```
void show_message (
            char * message,
            int type_msg )
```

This is the cli version of the one in gtkterm.

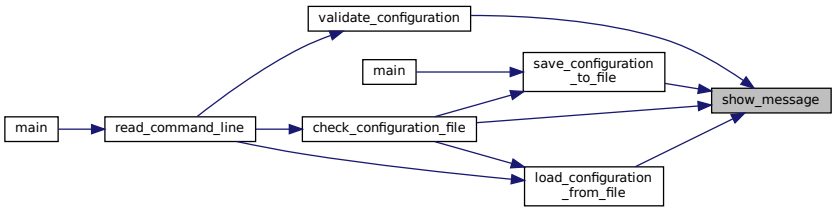
References **i18n_printf()**.

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

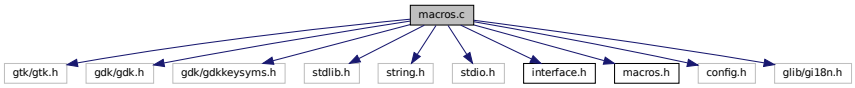
```
GtkWidget* Text  [extern]
```

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/gi18n.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.

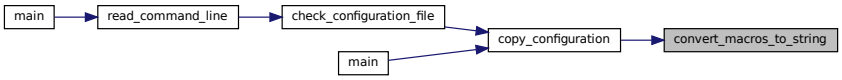
Must be NULL terminated

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()

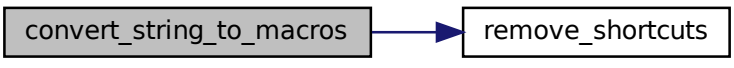
```
void convert_string_to_macros (
            char ** string_list,
            int size )
```

Convert the array of strings to macros.

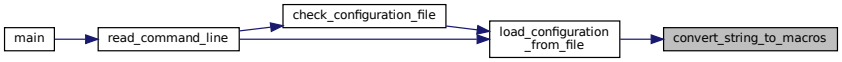
References **macro_t::action**, **macros**, **nr_of_macros**, **remove_shortcuts()**, and **macro_t::shortcut**.

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**.

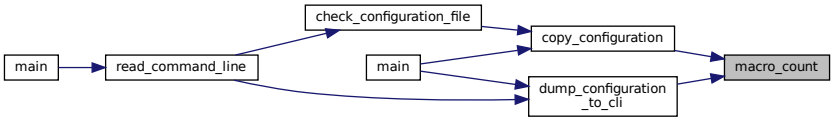
7.13.2.4 macro_count()

```
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()

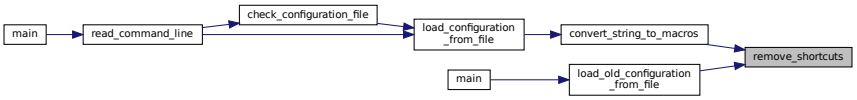
```
void remove_shortcuts (
            void  )
```

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()**.

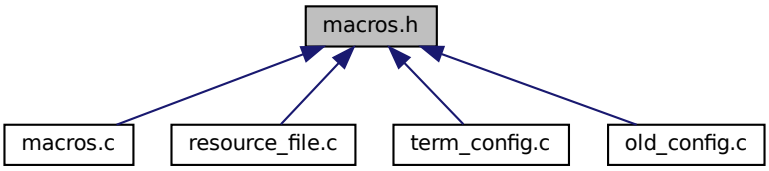
7.13.3.2 nr_of_macros

```
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)
- void **add_shortcuts** (void)

  Remove shortcuts from accel_group and free memory.
- **macro_t** * **get_shortcuts** (gint *)
- void **convert_string_to_macros** (char **, int)

  Convert the array of strings to macros.
- int **convert_macros_to_string** (char **)

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

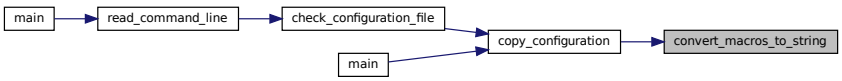
Must be NULL terminated

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()

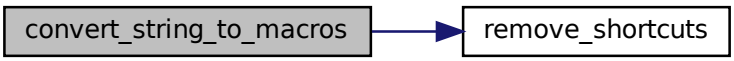
```
void convert_string_to_macros (
            char ** string_list,
            int size )
```

Convert the array of strings to macros.

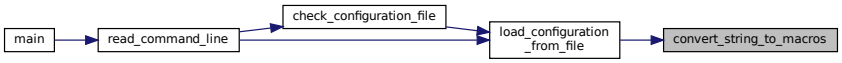
References **macro_t::action**, **macros**, **nr_of_macros**, **remove_shortcuts()**, and **macro_t::shortcut**.

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 *  )
```

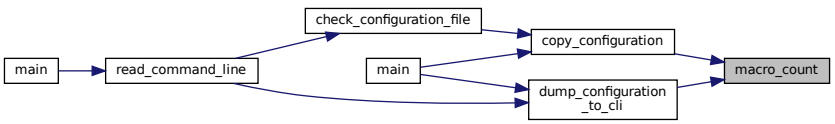
7.14.1.5 macro_count()

```
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()

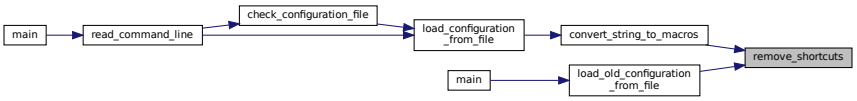
```
void remove_shortcuts (
            void  )
```

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**

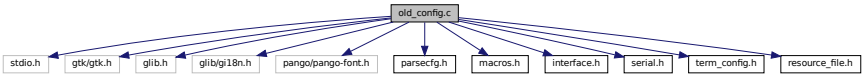
Variables

- **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/gi18n.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()

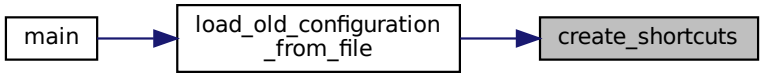
```
void create_shortcuts (
             macro_t * macro,
            int size )
```

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 )
```

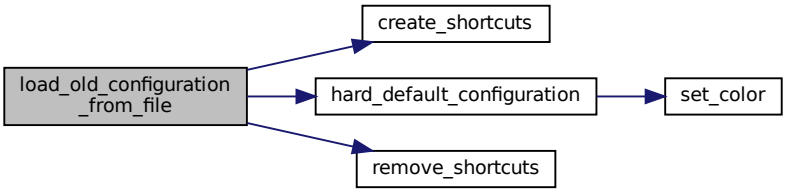
load old config file with parsecfg Because we convert all sections we can walk trough all section numbers

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**,

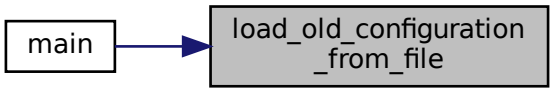
remove_shortcuts(), **rows**, **display_config_t::rows**, **port_config_t::rs485_rts_time_after_transmit**, **port_config_t::rs485_rts_time_before_transmit**, **rts_time_after_tx**, **rts_time_before_tx**, **scrollback**, **display_config_t::scrollback**, **macro_t::shortcut**, **speed**, **port_config_t::speed**, **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 scrollback

```
int* scrollback
```

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

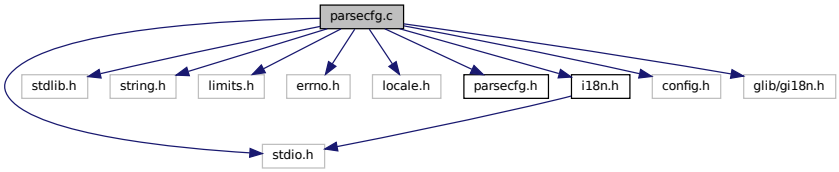
```
int* wait_delay
```

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()

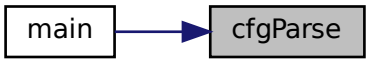
```
int cfgDump (
             const char * file,
              cfgStruct cfg[],
              cfgFileType type,
             int max_section )
```

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()

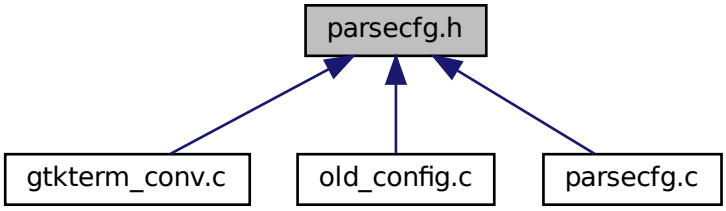
```
int cfgStoreValue (
             cfgStruct cfg[],
            const char * parameter,
            const char * value,
             cfgFileType type,
            int section )
```

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()

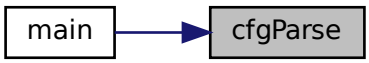
```
int cfgDump (
             const char * file,
              cfgStruct cfg[],
              cfgFileType type,
             int max_section )
```

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()

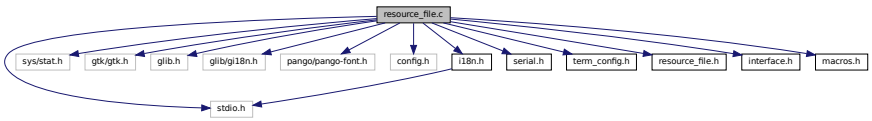
```
int fetchVarFromCfgFile (
            const char * file,
            char * parameter_name,
            void * result_value,
             cfgValueType value_type,
             cfgFileType file_type,
            int section_num,
            const char * section_name )
```

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/gi18n.h>
#include <pango/pango-font.h>
#include <config.h>
#include "i18n.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.

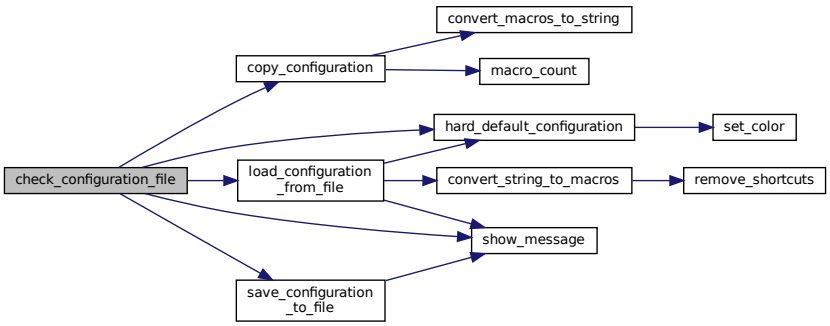
If not it creates a new [default] Put the new default in the key file

And save the config to file

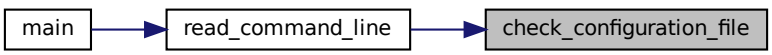
References **config**, **config_file**, **copy_configuration()**, **hard_default_configuration()**, **load_configuration_from_file()**, **MSG_WRN**, **save_configuration_to_file()**, and **show_message()**.

Referenced by **read_command_line()**.

Here is the call graph for this function:



Here is the caller graph for this function:



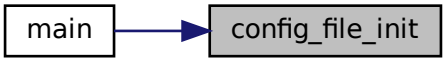
7.20.3.2 config_file_init()

```
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 )
```

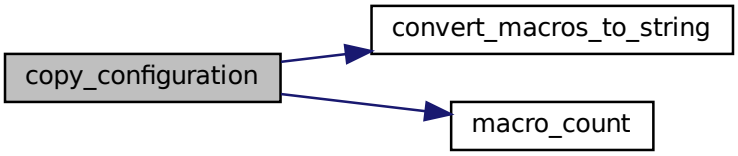
Copy the active configuration into <section> of the Key file.

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

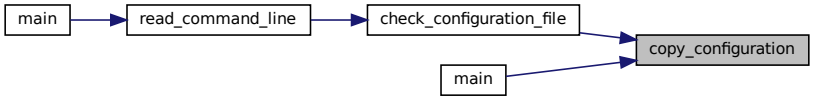
References **display_config_t::background_color**, **port_config_t::bits**, **display_config_t::char_queue**, **display_config_t::columns**, **CONF_ITEM_BITS**, **CONF_ITEM_CRLF_AUTO**, **CONF_ITEM_DISABLE↩ _PORT_LOCK**, **CONF_ITEM_ECHO**, **CONF_ITEM_FLOW_CONTROL**, **CONF_ITEM_FONT**, **CONF_↩ ITEM_MACROS**, **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

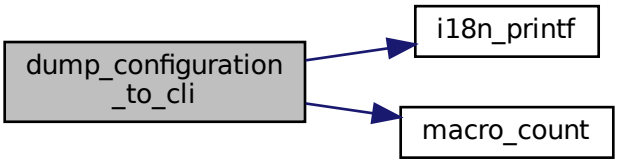
Print the terminal items

... and the macro's

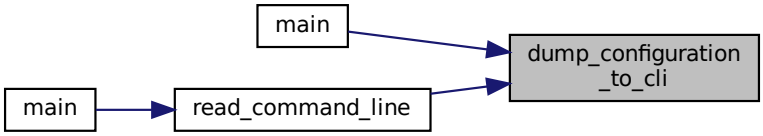
References **display_config_t::background_color**, **port_config_t::bits**, **display_config_t::block_cursor**, **display_config_t::char_queue**, **display_config_t::columns**, **display_config_t::crlfauto**, **display_config_t::delay**, **port_config_t::disable_port_lock**, **display_config_t::echo**, **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  )
```

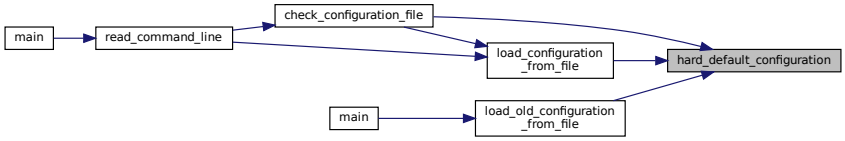
Create a new <default> configuration.

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.

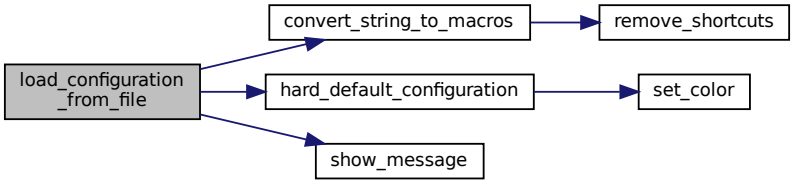
Convert the stringlist to macros. Existing shortcuts will be delete from convert_string_to_macros

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**,

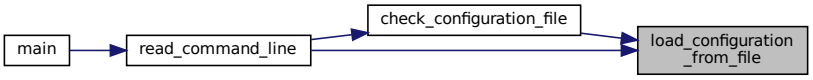
CONF_ITEM_WAIT_CHAR, **CONF_ITEM_WAIT_DELAY**, **config_file**, **ConfigurationItem**, **convert_↩ string_to_macros()**, **display_config_t::crlfauto**, **display_config_t::delay**, **port_config_t::disable_port↩ _lock**, **display_config_t::echo**, **port_config_t::flow_control**, **display_config_t::font**, **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.

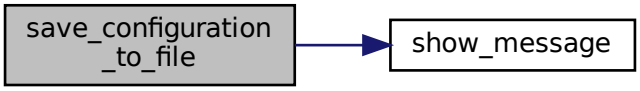
7.20.3.8 save_configuration_to_file()

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

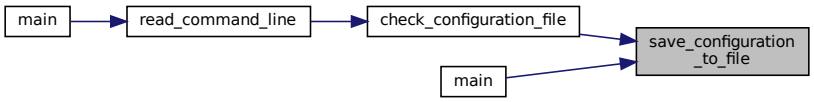
References **config**, **config_file**, **MSG_WRN**, and **show_message()**.

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

Here is the call graph for this function:



Here is the caller graph for this function:



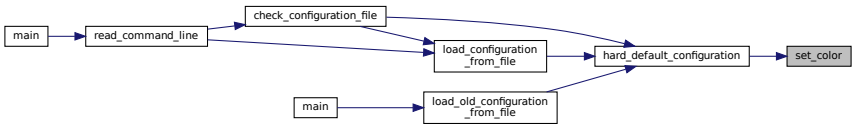
7.20.3.9 set_color()

```
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()

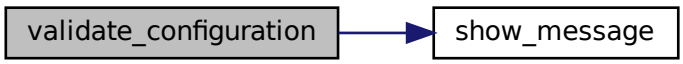
```
void validate_configuration (
            void  )
```

validate the active configuration

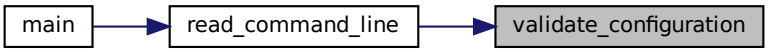
References **port_config_t::bits**, **DEFAULT_BITS**, **DEFAULT_DELAY**, **DEFAULT_FONT**, **DEFAULT_STOP**, **display_config_t::delay**, **display_config_t::font**, **MSG_ERR**, **port_conf**, **show_message()**, **port_config_t::speed**, **port_config_t::stops**, and **term_conf**.

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]
```

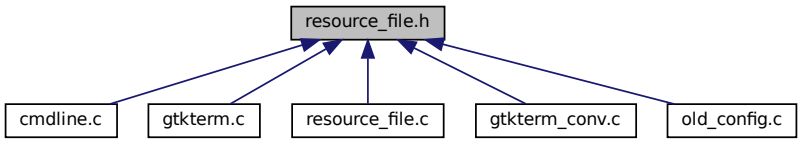
Initial value:

```
= {
        "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** ()

  This checks if the configuration file exists.
- void **dump_configuration_to_cli** (char *)
- void **hard_default_configuration** (void)

  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.

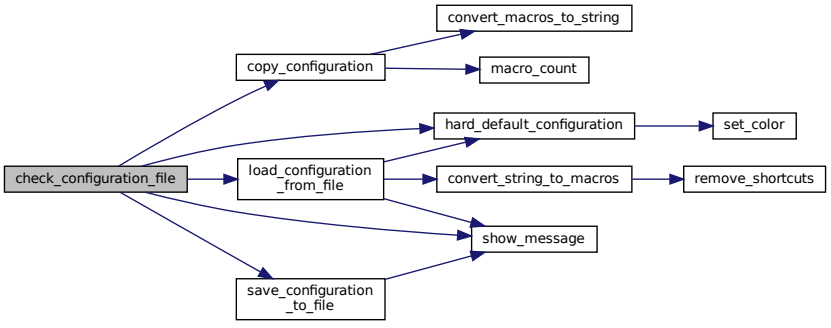
If not it creates a new [default] Put the new default in the key file

And save the config to file

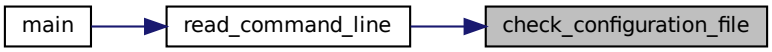
References **config**, **config_file**, **copy_configuration()**, **hard_default_configuration()**, **load_configuration↩ _from_file()**, **MSG_WRN**, **save_configuration_to_file()**, and **show_message()**.

Referenced by **read_command_line()**.

Here is the call graph for this function:



Here is the caller graph for this function:



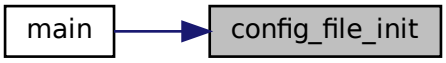
7.21.1.2 config_file_init()

```
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.

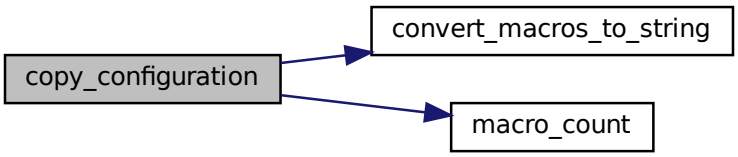
Macros are an array of strings, so we have to convert it All macros ends up in the string_list

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**,

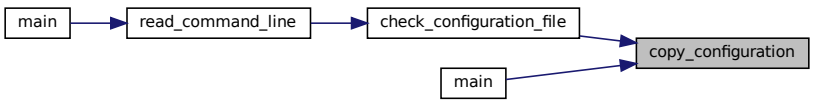
CONF_ITEM_RS485_RTS_TIME_BEFORE_TX, **CONF_ITEM_SPEED**, **CONF_ITEM_STOPBITS**, **CONF_↩ ITEM_TERM_BACKGROUND_ALPHA**, **CONF_ITEM_TERM_BACKGROUND_BLUE**, **CONF_ITEM_TERM_↩ BACKGROUND_GREEN**, **CONF_ITEM_TERM_BACKGROUND_RED**, **CONF_ITEM_TERM_COLS**, **CONF_↩ ITEM_TERM_FOREGROUND_ALPHA**, **CONF_ITEM_TERM_FOREGROUND_BLUE**, **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

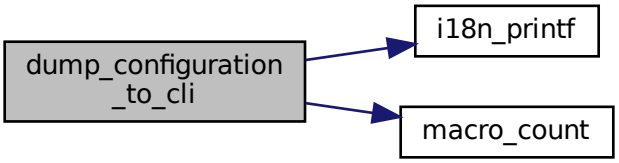
Print the terminal items

... and the macro's

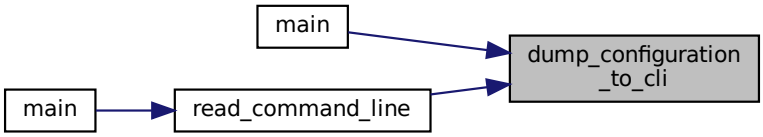
References **display_config_t::background_color**, **port_config_t::bits**, **display_config_t::block_cursor**, **display_config_t::char_queue**, **display_config_t::columns**, **display_config_t::crlfauto**, **display_config_t::delay**, **port_config_t::disable_port_lock**, **display_config_t::echo**, **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.

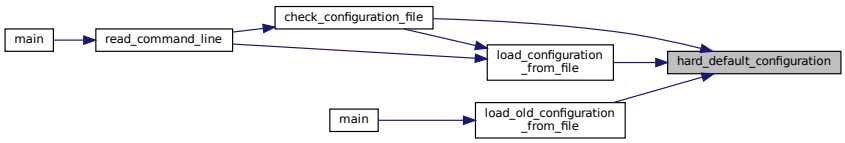
References **display_config_t::background_color**, **port_config_t::bits**, **display_config_t::block_cursor**, **display_config_t::char_queue**, **display_config_t::columns**, **display_config_t::crlfauto**, **DEFAULT_BITS**, **DEFAULT_CHAR**, **DEFAULT_DELAY**, **DEFAULT_DELAY_RS485**, **DEFAULT_ECHO**, **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.

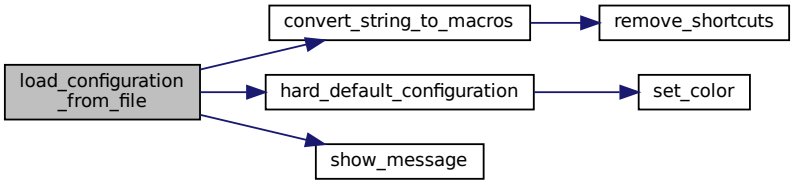
Convert the stringlist to macros. Existing shortcuts will be delete from convert_string_to_macros

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**,

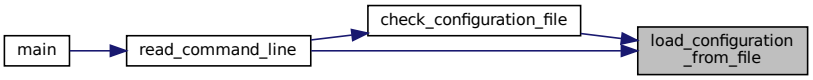
CONF_ITEM_RS485_RTS_TIME_BEFORE_TX, **CONF_ITEM_SPEED**, **CONF_ITEM_STOPBITS**, **CONF_ITEM_TERM_BACKGROUND_ALPHA**, **CONF_ITEM_TERM_BACKGROUND_BLUE**, **CONF_ITEM_TERM_BACKGROUND_GREEN**, **CONF_ITEM_TERM_BACKGROUND_RED**, **CONF_ITEM_TERM_COLS**, **CONF_ITEM_TERM_FOREGROUND_ALPHA**, **CONF_ITEM_TERM_FOREGROUND_BLUE**, **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.

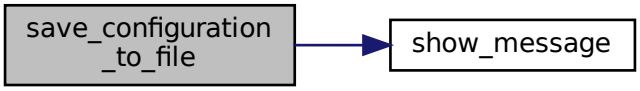
7.21.1.8 save_configuration_to_file()

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

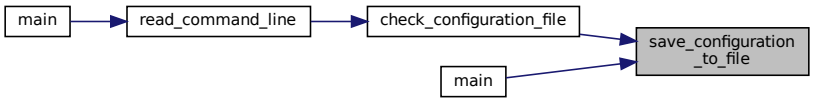
References **config**, **config_file**, **MSG_WRN**, and **show_message()**.

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

Here is the call graph for this function:



Here is the caller graph for this function:



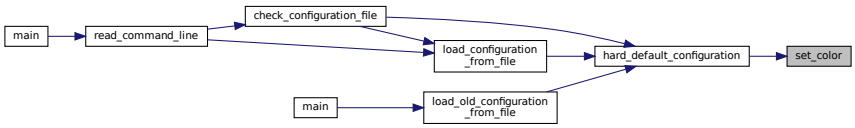
7.21.1.9 set_color()

```
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()

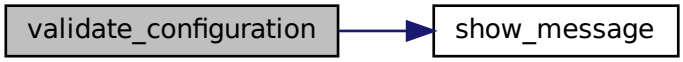
```
void validate_configuration (
            void  )
```

validate the active configuration

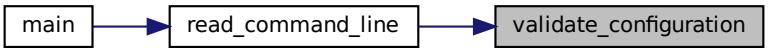
References **port_config_t::bits**, **DEFAULT_BITS**, **DEFAULT_DELAY**, **DEFAULT_FONT**, **DEFAULT_STOP**, **display_config_t::delay**, **display_config_t::font**, **MSG_ERR**, **port_conf**, **show_message()**, **port_config_t::speed**, **port_config_t::stops**, and **term_conf**.

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]
```

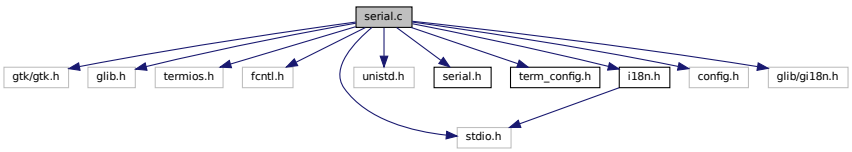
The key file.

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
```

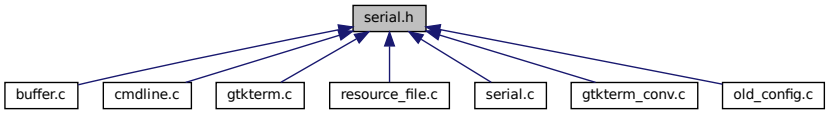
Referenced by **get_port_string()**.

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

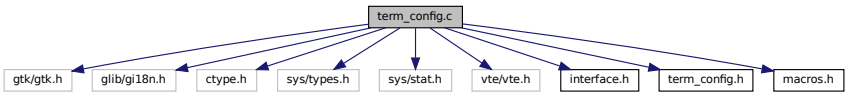
```
int serial_port_fd  [extern]
```

Referenced by **get_port_string()**.

7.24 term_config.c File Reference

```
#include <gtk/gtk.h>
#include <glib/gi18n.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

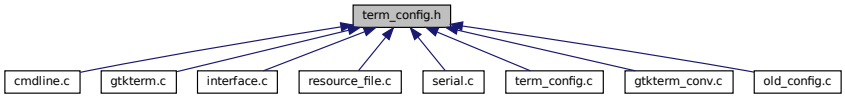
7.24.2.1 term_conf

```
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