

DTV App

Generated by Doxygen 1.8.11

Contents

1	Module Index	1
1.1	Modules	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Module Documentation	7
4.1	Configuration file interface	7
4.1.1	Detailed Description	7
4.1.2	Function Documentation	7
4.1.2.1	config_get_init_ch_info(FILE *f)	7
4.2	Drawing interface	8
4.2.1	Detailed Description	8
4.3	DTV interface	9
4.3.1	Detailed Description	9
4.3.2	Function Documentation	9
4.3.2.1	dtv_set_volume(uint8_t vol)	9
4.4	Graphics interface	10
4.4.1	Detailed Description	10
4.4.2	Function Documentation	10
4.4.2.1	graphics_show_volume(uint8_t vol)	10
4.5	Table parsing interface	12
4.5.1	Detailed Description	12
4.6	Remote-control interface	13
4.6.1	Detailed Description	13
4.6.2	Function Documentation	13
4.6.2.1	rc_start_loop(const char *dev, rc_key_callback callback)	13
4.7	DVB table retrieval interface	14
4.7.1	Detailed Description	15

5	Data Structure Documentation	17
5.1	config_init_ch_info Struct Reference	17
5.1.1	Detailed Description	17
5.2	draw_interface Struct Reference	18
5.2.1	Detailed Description	18
5.3	dtv_channel_info Struct Reference	18
5.3.1	Detailed Description	19
5.4	graphics_args Struct Reference	19
5.4.1	Detailed Description	19
5.5	graphics_channel_info Struct Reference	19
5.5.1	Detailed Description	20
5.6	graphics_flags Struct Reference	20
5.6.1	Detailed Description	21
5.7	pat Struct Reference	21
5.7.1	Detailed Description	22
5.8	pat_body Struct Reference	22
5.8.1	Detailed Description	22
5.9	pat_header Struct Reference	22
5.9.1	Detailed Description	23
5.10	pmt Struct Reference	23
5.10.1	Detailed Description	23
5.11	pat::pmt_basic Struct Reference	24
5.11.1	Detailed Description	24
5.12	pmt_body Struct Reference	24
5.12.1	Detailed Description	25
5.13	pmt_header Struct Reference	25
5.13.1	Detailed Description	26
5.14	rc_args Struct Reference	26
5.14.1	Detailed Description	26
5.15	sdt Struct Reference	26

5.15.1 Detailed Description	27
5.16 sdt_body Struct Reference	27
5.16.1 Detailed Description	27
5.17 sdt_descriptor1 Struct Reference	27
5.17.1 Detailed Description	28
5.18 sdt_descriptor2 Struct Reference	28
5.18.1 Detailed Description	28
5.19 sdt_header Struct Reference	29
5.19.1 Detailed Description	29
5.20 table_header Struct Reference	29
5.20.1 Detailed Description	30
5.21 teletext_descriptor_header Struct Reference	30
5.21.1 Detailed Description	30
5.22 tot_descriptor_body Struct Reference	31
5.22.1 Detailed Description	31
5.23 tot_descriptor_header Struct Reference	31
5.23.1 Detailed Description	31
5.24 tot_header Struct Reference	32
5.24.1 Detailed Description	32
6 File Documentation	33
6.1 include/common.h File Reference	33
6.1.1 Detailed Description	34
6.1.2 Macro Definition Documentation	34
6.1.2.1 FAIL	34
6.1.2.2 FAIL_STD	34
6.1.2.3 LOG	34
6.2 include/config.h File Reference	35
6.2.1 Detailed Description	36
6.3 include/drawing.h File Reference	36
6.3.1 Detailed Description	38

6.4	include/dtv.h File Reference	38
6.4.1	Detailed Description	39
6.5	include/graphics.h File Reference	40
6.5.1	Detailed Description	41
6.6	include/parsing.h File Reference	41
6.6.1	Detailed Description	42
6.7	include/rc.h File Reference	43
6.7.1	Detailed Description	44
6.8	include/structures.h File Reference	44
6.8.1	Detailed Description	47
6.8.2	Variable Documentation	47
6.8.2.1	ch_num	47
6.8.2.2	len	47
6.8.2.3	pid	47
6.8.2.4	type	47
6.9	src/config.c File Reference	48
6.9.1	Detailed Description	49
6.9.2	Macro Definition Documentation	49
6.9.2.1	BUF_SIZE	49
6.9.2.2	MAKE_GETTER	49
6.10	src/graphics.c File Reference	50
6.10.1	Detailed Description	51
6.10.2	Macro Definition Documentation	51
6.10.2.1	DRAWCHECK	51
6.10.3	Enumeration Type Documentation	52
6.10.3.1	g_error	52
6.11	src/main.c File Reference	52
6.11.1	Detailed Description	53
6.12	src/rc.c File Reference	53
6.12.1	Detailed Description	53

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Configuration file interface	7
Drawing interface	8
DTV interface	9
Graphics interface	10
Table parsing interface	12
Remote-control interface	13
DVB table retrieval interface	14

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

config_init_ch_info	A structure that holds the initial dtv settings	17
draw_interface	Basic interface necessary to display graphics elements	18
dtv_channel_info	Contains basic channel info	18
graphics_args	A shim structure to pass main arguments to DirectFB	19
graphics_channel_info	A struct that contains some basic channel info	19
graphics_flags	A struct that keeps the state of what should be displayed	20
pat	Contains important info from the PAT table	21
pat_body	Represents PAT body	22
pat_header	Represents PAT header	22
pmt	Contains important info from the PMT table	23
pat::pmt_basic	Contains enough info to identify a PMT table	24
pmt_body	Represents PMT body	24
pmt_header	Represents PMT header	25
rc_args	A shim structure to pass arguments to event thread	26
sdt	Contains important info from the SDT table	26
sdt_body	Represents SDT body	27
sdt_descriptor1	Represents the first half of the service descriptor	27
sdt_descriptor2	Represents the second half of the service descriptor	28

sdt_header	Represents SDT header	29
table_header	Header that is part of all tables	29
teletext_descriptor_header	Represents the teletext descriptor header	30
tot_descriptor_body	Represents TOT descriptor body	31
tot_descriptor_header	Represents TOT descriptor header	31
tot_header	Represents TOT header	32

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

include/common.h	Contains functions that all modules use	33
include/config.h	Contains configuration file API	35
include/drawing.h	Contains API for drawing graphics elements	36
include/dtv.h	Contains DTV API	38
include/graphics.h	Contains the graphics API	40
include/parsing.h	Contains streamlined internal representations of tables	41
include/rc.h	Contains Remote Control API	43
include/structures.h	Contains nitty-gritty DVB details	44
src/config.c	Implementation for the configuration file interface	48
src/graphics.c	Contains implementation for graphics interface	50
src/main.c	Contains the implementation that glues other modules together	52
src/rc.c	Contains the implementation of the remote control interface	53

Chapter 4

Module Documentation

4.1 Configuration file interface

Functions and structures for retrieving configuration options.

Data Structures

- struct `config_init_ch_info`
A structure that holds the initial dtv settings.

Functions

- struct `config_init_ch_info` `config_get_init_ch_info` (FILE *f)
Reads the initial settings from the specified file.

4.1.1 Detailed Description

Functions and structures for retrieving configuration options.

4.1.2 Function Documentation

4.1.2.1 struct `config_init_ch_info` `config_get_init_ch_info` (FILE * f)

Reads the initial settings from the specified file.

Reads the initial settings from the specified file.

4.2 Drawing interface

Functions and structures for drawing.

Data Structures

- struct [draw_interface](#)
Basic interface necessary to display graphics elements.

Functions

- `int32_t draw_init (struct draw_interface *draw_i, int *argc, char ***argv)`
Initialize drawing interface.
- `int32_t draw_channel_info (struct draw_interface *draw_i, struct graphics_channel_info info)`
Draw channel_info graphics element.
- `int32_t draw_init_message (struct draw_interface *draw_i)`
Draw initializing message.
- `int32_t draw_time (struct draw_interface *draw_i, struct tm tm)`
Draw time graphics element.
- `int32_t draw_volume (struct draw_interface *draw_i, uint8_t vol)`
Draw volume graphics element.
- `int32_t draw_no_channel (struct draw_interface *draw_i)`
Draw no channel display.
- `int32_t draw_audio_only (struct draw_interface *draw_i)`
Draw radio graphics.
- `int32_t draw_channel_number (struct draw_interface *draw_i, uint16_t ch_num)`
Draw channel number.
- `int32_t draw_blackscreen (struct draw_interface *draw_i)`
Draw a black rectangle.
- `int32_t draw_clear (struct draw_interface *draw_i)`
Clear the screen.
- `int32_t draw_refresh (struct draw_interface *draw_i)`
Refresh display.
- `int32_t draw_deinit (struct draw_interface *draw_i)`
Deinitialize drawing interface.

4.2.1 Detailed Description

Functions and structures for drawing.

4.3 DTV interface

Functions and structures for controlling DTV functionality.

Data Structures

- struct [dtv_channel_info](#)
Contains basic channel info.

Functions

- void [dtv_init](#) (struct [config_init_ch_info](#) init_info)
Function that initializes internal DTV state.
- struct [dtv_channel_info](#) [dtv_switch_channel](#) (uint16_t ch_num)
Tries to switch to the desired channel.
- t_Error [dtv_set_volume](#) (uint8_t vol)
Tries to set the volume to the desired value.
- struct tm [dtv_get_time](#) ()
Gets the time information.
- struct [sdt_dtv_get_info](#) (uint16_t ch_num)
Gets the SDT information for the specified channel.
- void [dtv_deinit](#) ()
Deinitializes the internal DTV state.

4.3.1 Detailed Description

Functions and structures for controlling DTV functionality.

4.3.2 Function Documentation

4.3.2.1 t_Error dtv_set_volume (uint8_t vol)

Tries to set the volume to the desired value.

Parameters

<i>vol</i>	Desired volume, should be [0-10].
------------	-----------------------------------

4.4 Graphics interface

Functions and structures for graphics interaction.

Data Structures

- struct [graphics_channel_info](#)
A struct that contains some basic channel info.

Functions

- void [graphics_show_channel_info](#) (struct [graphics_channel_info](#) info)
Displays some basic information about a channel on the screen.
- void [graphics_show_init](#) ()
Displays initializing message.
- void [graphics_hide_init](#) ()
Removes initializing message.
- void [graphics_show_time](#) (struct tm tm)
Displays current time.
- void [graphics_show_volume](#) (uint8_t vol)
Displays volume information on the screen.
- void [graphics_show_mute](#) ()
Displays mute symbol.
- void [graphics_hide_mute](#) ()
Removes mute symbol.
- void [graphics_show_channel_number](#) (uint16_t ch_num)
Displays selected channel number.
- void [graphics_blackscreen](#) ()
Puts a black screen on the screen.
- void [graphics_clear](#) ()
Clears all graphics elements from screen.
- void [graphics_start_render](#) (int *argc, char ***argv)
Starts rendering graphic elements on screen.
- void [graphics_stop](#) ()
Stops graphics rendering loop.

4.4.1 Detailed Description

Functions and structures for graphics interaction.

4.4.2 Function Documentation

4.4.2.1 void [graphics_show_volume](#) (uint8_t vol)

Displays volume information on the screen.

Parameters

<i>vol</i>	Must be [0-10].
------------	-----------------

4.5 Table parsing interface

Functions and structures for parsing DVB tables into internal form.

Data Structures

- struct [pat](#)
Contains important info from the PAT table.
- struct [pmt](#)
Contains important info from the PMT table.
- struct [sdt](#)
Contains important info from the SDT table.

Functions

- struct [pat](#) [parse_pat](#) (const uint8_t *buffer)
Parses the given buffer as a PAT table.
- struct [pmt](#) [parse_pmt](#) (const uint8_t *buffer)
Parses the given buffer as a PMT table.
- struct tm [parse_tot](#) (const uint8_t *buffer)
Parses the given buffer as a TOT table and converts its info to C representation of time.
- struct [sdt](#) [parse_sdt](#) (const uint8_t *buffer, uint16_t [ch_num](#))
Parses the given buffer as a SDT table and extracts information about the specified channel.

4.5.1 Detailed Description

Functions and structures for parsing DVB tables into internal form.

4.6 Remote-control interface

Functions and structures for remote control interaction.

Typedefs

- typedef void(* [rc_key_callback](#)) (int key_no)
A callback that should take action on key press.

Functions

- void [rc_start_loop](#) (const char *dev, [rc_key_callback](#) callback)
A function that starts the loop that waits for input events from the remote control.
- void [rc_stop_loop](#) ()
Stops the event loop.

4.6.1 Detailed Description

Functions and structures for remote control interaction.

4.6.2 Function Documentation

4.6.2.1 void [rc_start_loop](#) (const char * *dev*, [rc_key_callback](#) *callback*)

A function that starts the loop that waits for input events from the remote control.

Parameters

<i>dev</i>	Name of the device to capture events from.
------------	--

4.7 DVB table retrieval interface

These functions retrieve the corresponding structures from the stream, performing the needed network-to-host conversions.

Data Structures

- struct [table_header](#)
Header that is part of all tables.
- struct [pat_header](#)
Represents PAT header.
- struct [pat_body](#)
Represents PAT body.
- struct [pmt_header](#)
Represents PMT header.
- struct [pmt_body](#)
Represents PMT body.
- struct [teletext_descriptor_header](#)
Represents the teletext descriptor header.
- struct [sdt_header](#)
Represents SDT header.
- struct [sdt_body](#)
Represents SDT body.
- struct [sdt_descriptor1](#)
Represents the first half of the service descriptor.
- struct [sdt_descriptor2](#)
Represents the second half of the service descriptor.
- struct [tot_header](#)
Represents TOT header.
- struct [tot_descriptor_header](#)
Represents TOT descriptor header.
- struct [tot_descriptor_body](#)
Represents TOT descriptor body.

Functions

- struct [table_header](#) **__attribute__((packed))**
- struct [pat_header](#) [get_pat_header](#) (const uint8_t *buffer)
Retrieves [pat_header](#) from the stream.
- struct [pat_body](#) [get_pat_body](#) (const uint8_t *buffer)
Retrieves [pat_body](#) from the stream.
- struct [pmt_header](#) [get_pmt_header](#) (const uint8_t *buffer)
Retrieves [pmt_header](#) from the stream.
- struct [pmt_body](#) [get_pmt_body](#) (const uint8_t *buffer)
Retrieves [pmt_body](#) from the stream.
- struct [teletext_descriptor_header](#) [get_teletext_descriptor_header](#) (const uint8_t *buffer)
Retrieves [teletext_descriptor_header](#) from the stream.
- struct [sdt_header](#) [get_sdt_header](#) (const uint8_t *buffer)
Retrieves [sdt_header](#) from the stream.

- struct `sdt_body` `get_sdt_body` (const uint8_t *buffer)
Retrieves `sdt_body` from the stream.
- struct `sdt_descriptor1` `get_sdt_descriptor1` (const uint8_t *buffer)
Retrieves `sdt_descriptor1` from the stream.
- struct `sdt_descriptor2` `get_sdt_descriptor2` (const uint8_t *buffer)
Retrieves `sdt_descriptor2` from the stream.
- struct `tot_header` `get_tot_header` (const uint8_t *buffer)
Retrieves `tot_header` from the stream.
- struct `tot_descriptor_header` `get_tot_descriptor_header` (const uint8_t *buffer)
Retrieves `tot_descriptor_header` from the stream.
- struct `tot_descriptor_body` `get_tot_descriptor_body` (const uint8_t *buffer)
Retrieves `tot_descriptor_body` from the stream.

Variables

- struct `teletext_descriptor_header` `__attribute__`

4.7.1 Detailed Description

These functions retrieve the corresponding structures from the stream, performing the needed network-to-host conversions.

Chapter 5

Data Structure Documentation

5.1 config_init_ch_info Struct Reference

A structure that holds the initial dtv settings.

```
#include <config.h>
```

Data Fields

- uint32_t [freq](#)
Tuner frequency.
- uint32_t [bandwidth](#)
Tuner bandwidth.
- enum t_Module [module](#)
Whether the channel uses DTB-T or DTB-T2.
- uint16_t [vpid](#)
pid of the initial video stream.
- uint16_t [apid](#)
pid of the initial audio stream.
- enum t_StreamType [vtype](#)
type of the initial video stream.
- enum t_StreamType [atype](#)
type of the initial audio stream.
- uint32_t [ch_num](#)
Channel number.
- int [teletext](#)
Whether the channel has teletext.

5.1.1 Detailed Description

A structure that holds the initial dtv settings.

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

- include/[config.h](#)

5.2 draw_interface Struct Reference

Basic interface necessary to display graphics elements.

```
#include <drawing.h>
```

Data Fields

- IDirectFBSurface * [surface](#)
Surface on which to draw.
- IDirectFB * [dfb_interface](#)
Main DFB interface.
- int32_t [screen_width](#)
The width of the screen.
- int32_t [screen_height](#)
The height of the screen.
- IDirectFBSurface * [vol_surfaces](#) [12]
Preloaded volume images.
- IDirectFBFont * [font_interface](#)
Preloaded font.

5.2.1 Detailed Description

Basic interface necessary to display graphics elements.

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

- include/[drawing.h](#)

5.3 dtv_channel_info Struct Reference

Contains basic channel info.

```
#include <dtv.h>
```

Data Fields

- uint16_t [ch_num](#)
Channel number.
- uint16_t [vpid](#)
PID of the video stream.
- uint16_t [apid](#)
PID of the audio stream.
- bool [teletext](#)
Specifies whether the channel has teletext.

5.3.1 Detailed Description

Contains basic channel info.

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

- [include/dtv.h](#)

5.4 graphics_args Struct Reference

A shim structure to pass main arguments to DirectFB.

Data Fields

- `int * argcx`
- `char *** argvx`

5.4.1 Detailed Description

A shim structure to pass main arguments to DirectFB.

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

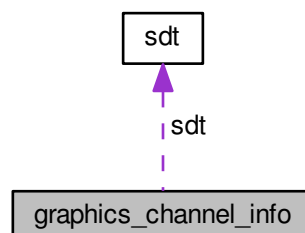
- [src/graphics.c](#)

5.5 graphics_channel_info Struct Reference

A struct that contains some basic channel info.

```
#include <graphics.h>
```

Collaboration diagram for graphics_channel_info:



Data Fields

- uint16_t [ch_num](#)
The number of the channel.
- bool [teletext](#)
Whether the channel has teletext.
- uint16_t [vpid](#)
The video PID of the channel.
- uint16_t [apid](#)
The audio PID of the channel.
- struct [sdt sdt](#)
Channel type and name.
- struct tm [tm](#)
Time information.

5.5.1 Detailed Description

A struct that contains some basic channel info.

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

- [include/graphics.h](#)

5.6 [graphics_flags](#) Struct Reference

A struct that keeps the state of what should be displayed.

Data Fields

- bool [info](#)
Specifies whether the info panel should be displayed.
- bool [volume](#)
Specifies whether the volume panel should be displayed.
- bool [blackscreen](#)
Specifies whether the screen should be filled with black.
- bool [no_channel](#)
Specifies whether the "NO CHANNEL" message should be displayed.
- bool [audio_only](#)
Specifies whether the "AUDIO ONLY" message should be displayed.
- bool [ch_num](#)
Specifies whether the top left channel number should be displayed.
- bool [time](#)
Specifies whether the time should be displayed.
- bool [init](#)
Specifies whether the "INITIALIZING" message should be displayed.
- bool [mute](#)
Specifies whether the mute symbol should be displayed. Takes precedence over [volume](#).

5.6.1 Detailed Description

A struct that keeps the state of what should be displayed.

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

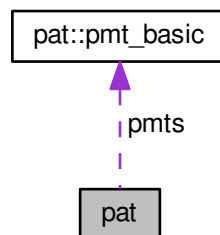
- [src/graphics.c](#)

5.7 pat Struct Reference

Contains important info from the PAT table.

```
#include <parsing.h>
```

Collaboration diagram for pat:



Data Structures

- struct [pmt_basic](#)
Contains enough info to identify a PMT table.

Data Fields

- [uint16_t](#) [tsi](#)
Transport stream id.
- [size_t](#) [pmt_len](#)
How many PMTs exist in the stream.
- struct [pat::pmt_basic](#) * [pmts](#)
Array of PMTs that exist in the stream.

5.7.1 Detailed Description

Contains important info from the PAT table.

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

- [include/parsing.h](#)

5.8 pat_body Struct Reference

Represents PAT body.

```
#include <structures.h>
```

Data Fields

- uint16_t [ch_num](#)
Channel number of the current PMT.
- union {
 - struct {
 - uint16_t [pid](#): 13
PID of the PMT table.
 - uint16_t [res](#): 3
 - b1s**
 - uint16_t [bitfield1](#)
- b1u**

5.8.1 Detailed Description

Represents PAT body.

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

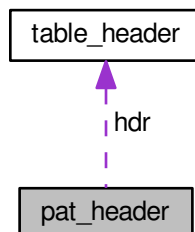
- [include/structures.h](#)

5.9 pat_header Struct Reference

Represents PAT header.

```
#include <structures.h>
```

Collaboration diagram for pat_header:



Data Fields

- struct [table_header](#) **hdr**
- uint16_t **tsi**
- struct {
 - uint8_t **cni**: 1
 - uint8_t **version**: 5
 - uint8_t **res**: 2
- **b1s**
- uint8_t **sec**
- uint8_t **lsn**

5.9.1 Detailed Description

Represents PAT header.

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

- include/[structures.h](#)

5.10 pmt Struct Reference

Contains important info from the PMT table.

```
#include <parsing.h>
```

Data Fields

- uint16_t [pid](#)
The pid of the PMT table.
- uint16_t [ch_num](#)
The channel number of the PMT table.
- uint16_t [video_pid](#)
PID of the video stream. It is -1 if it doesn't exist.
- uint16_t [audio_pid](#)
PID of the audio stream. It is -1 if it doesn't exist.
- bool [teletext](#)
Specifies whether the channel has teletext.

5.10.1 Detailed Description

Contains important info from the PMT table.

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

- include/[parsing.h](#)

5.11 pat::pmt_basic Struct Reference

Contains enough info to identify a PMT table.

```
#include <parsing.h>
```

Data Fields

- uint16_t [pid](#)
The PID of the PMT table.
- uint16_t [ch_num](#)
The channel number of the PMT table.

5.11.1 Detailed Description

Contains enough info to identify a PMT table.

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

- include/[parsing.h](#)

5.12 pmt_body Struct Reference

Represents PMT body.

```
#include <structures.h>
```

Data Fields

- uint8_t [type](#)
Type of stream.
- union {
 struct {
 uint16_t [pid](#): 13
PID of the PS.
 uint16_t [res](#): 3
 } **b1s**
 uint16_t [bitfield1](#)
 } **b1u**
- union {
 struct {
 uint16_t [esilen](#): 12
Length of the descriptor section.
 uint16_t [res2](#): 4
 } **b2s**
 uint16_t [bitfield2](#)
 } **b2u**

5.12.1 Detailed Description

Represents PMT body.

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

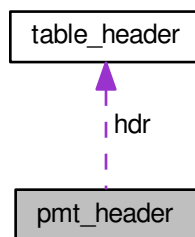
- include/[structures.h](#)

5.13 pmt_header Struct Reference

Represents PMT header.

```
#include <structures.h>
```

Collaboration diagram for pmt_header:



Data Fields

- struct [table_header](#) **hdr**
- uint16_t [ch_num](#)
Channel number.
- struct {
 uint8_t **cni**: 1
 uint8_t **version**: 5
 uint8_t **res**: 2
} **b**
- uint8_t **sec**
- uint8_t **lsn**
- union {
 struct {
 uint16_t **pcr_pid**: 13
 uint16_t **res2**: 3
 } **b1s**
 uint16_t **bitfield1**
} **b1u**

- union {
 - struct {
 - uint16_t **pilen**: 12
*Length of the **pmt_body** section.*
 - uint16_t **res3**: 4
 - b2s**
 - uint16_t **bitfield2**
- b2u**

5.13.1 Detailed Description

Represents PMT header.

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

- include/[structures.h](#)

5.14 rc_args Struct Reference

A shim structure to pass arguments to event thread.

Data Fields

- int **fd**
- [rc_key_callback](#) **kc**

5.14.1 Detailed Description

A shim structure to pass arguments to event thread.

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

- src/[rc.c](#)

5.15 sdt Struct Reference

Contains important info from the SDT table.

```
#include <parsing.h>
```

Data Fields

- uint8_t **st**
Service type.
- char **name** [100]
Channel name.

5.15.1 Detailed Description

Contains important info from the SDT table.

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

- [include/parsing.h](#)

5.16 sdt_body Struct Reference

Represents SDT body.

```
#include <structures.h>
```

Data Fields

- `uint16_t sid`
Service id (same as channel number).
- `struct {`
 - `uint8_t epff: 1`
 - `uint8_t esf: 1`
 - `uint8_t res: 6` `} b1s`
- `union {`
 - `struct {`
 - `uint16_t dlen: 12`
Length of the descriptor section.
 - `uint16_t fcm: 1`
 - `uint16_t rs: 3`
 - `} b2s`
 - `uint16_t bitfield2` `} b2u`

5.16.1 Detailed Description

Represents SDT body.

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

- [include/structures.h](#)

5.17 sdt_descriptor1 Struct Reference

Represents the first half of the service descriptor.

```
#include <structures.h>
```

Data Fields

- `uint8_t tag`
- `uint8_t len`
Length of the descriptor.
- `uint8_t type`
Type of service (channel).
- `uint8_t spnlen`
Length of the service provider name.

5.17.1 Detailed Description

Represents the first half of the service descriptor.

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

- `include/structures.h`

5.18 sdt_descriptor2 Struct Reference

Represents the second half of the service descriptor.

```
#include <structures.h>
```

Data Fields

- `uint8_t snlen`
Length of the service name.

5.18.1 Detailed Description

Represents the second half of the service descriptor.

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

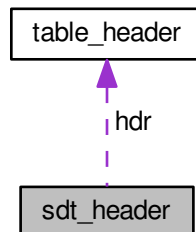
- `include/structures.h`

5.19 sdt_header Struct Reference

Represents SDT header.

```
#include <structures.h>
```

Collaboration diagram for sdt_header:



Data Fields

- struct [table_header](#) **hdr**
- uint16_t **tsi**
- struct {
 - uint8_t **cni**: 1
 - uint8_t **version**: 5
 - uint8_t **res**: 2
- **b1s**
- uint8_t **sec**
- uint8_t **lsn**
- uint16_t **oni**
- uint8_t **res2**

5.19.1 Detailed Description

Represents SDT header.

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

- include/[structures.h](#)

5.20 table_header Struct Reference

Header that is part of all tables.

```
#include <structures.h>
```

Data Fields

- `uint8_t tid`
Table id.
- `union {`
 - `struct {`
 - `uint16_t len: 12`
Length of the rest of the table.
 - `uint16_t res: 2`
 - `uint16_t zero: 1`
 - `uint16_t ssi: 1`
 - `} b1s`
 - `uint16_t bitfield1`
- `} b1u`

5.20.1 Detailed Description

Header that is part of all tables.

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

- `include/structures.h`

5.21 teletext_descriptor_header Struct Reference

Represents the teletext descriptor header.

```
#include <structures.h>
```

Data Fields

- `uint8_t tag`
Tag, should be 0x56.
- `uint8_t len`
Length of the descriptor.

5.21.1 Detailed Description

Represents the teletext descriptor header.

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

- `include/structures.h`

5.22 tot_descriptor_body Struct Reference

Represents TOT descriptor body.

```
#include <structures.h>
```

Data Fields

- union {
 - struct {
 - uint32_t **cc**: 24
 - uint32_t **regid**: 6
 - uint32_t **res**: 1
 - uint32_t **pol**: 1
 - } **b1s**
 - uint32_t **bitfield1**
- } **b1u**
- uint16_t **lto**
Local time offset.
- uint8_t **toc** [5]
- uint16_t **nto**

5.22.1 Detailed Description

Represents TOT descriptor body.

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

- include/[structures.h](#)

5.23 tot_descriptor_header Struct Reference

Represents TOT descriptor header.

```
#include <structures.h>
```

Data Fields

- uint8_t **tag**
- uint8_t **len**
Length of the descriptor body.

5.23.1 Detailed Description

Represents TOT descriptor header.

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

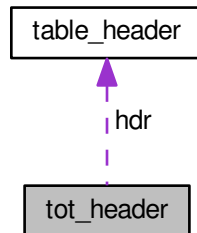
- include/[structures.h](#)

5.24 tot_header Struct Reference

Represents TOT header.

```
#include <structures.h>
```

Collaboration diagram for tot_header:



Data Fields

- struct [table_header](#) **hdr**
- `uint8_t` [time](#) [5]
Brain-dead encoded time information.
- union {
 - struct {
 - `uint16_t` [dlen](#): 12
Length of the descriptor section.
 - `uint16_t` **res**: 4
 - } **b1s**
 - `uint16_t` **bitfield1**
- } **b1u**

5.24.1 Detailed Description

Represents TOT header.

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

- `include/structures.h`

Chapter 6

File Documentation

6.1 include/common.h File Reference

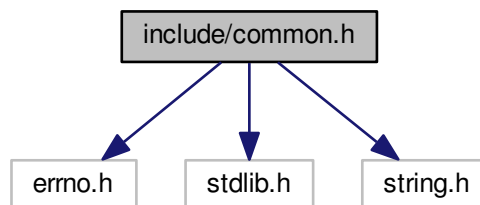
Contains functions that all modules use.

```
#include <errno.h>
```

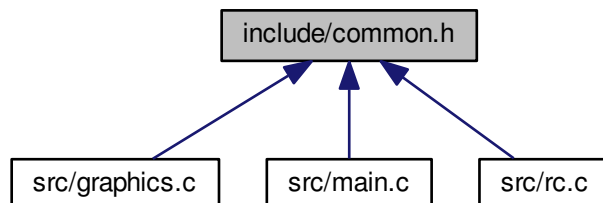
```
#include <stdlib.h>
```

```
#include <string.h>
```

Include dependency graph for common.h:



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



Macros

- `#define nameof(x) #x`
Gets the string representation of its parameter.
- `#define FAIL(fmt, ...)`
Prints an error message to stderr and exits the program.
- `#define FAIL_STD(fmt, ...) FAIL("%s: "fmt, strerror(errno), ##__VA_ARGS__)`
Same as [FAIL](#), except that it also prints the string representation of errno.
- `#define LOG(module, fmt, ...)`
Prints out a log message.

6.1.1 Detailed Description

Contains functions that all modules use.

6.1.2 Macro Definition Documentation

6.1.2.1 `#define FAIL(fmt, ...)`

Value:

```
do { \
    fprintf(stderr, "%s:%d:%s: Error: "fmt, \
        __FILE__, __LINE__, __func__, ##__VA_ARGS__); \
    exit(EXIT_FAILURE); \
} while (0)
```

Prints an error message to stderr and exits the program.

Parameters

<i>fmt</i>	A printf-like format string.
...	printf-like arguments to print.

6.1.2.2 `#define FAIL_STD(fmt, ...) FAIL("%s: "fmt, strerror(errno), ##__VA_ARGS__)`

Same as [FAIL](#), except that it also prints the string representation of errno.

Parameters

<i>fmt</i>	A printf-like format string.
...	printf-like arguments to print.

6.1.2.3 `#define LOG(module, fmt, ...)`

Value:


```
do \
{ \
    printf(module": "fmt, ##__VA_ARGS__); \
} while (0)
```

Prints out a log message.

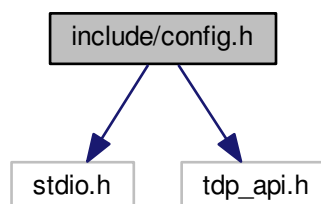
Parameters

<i>module</i>	A string that identifies the module in which LOG is called.
<i>fmt</i>	A printf-like format string.
...	printf-like arguments to print.

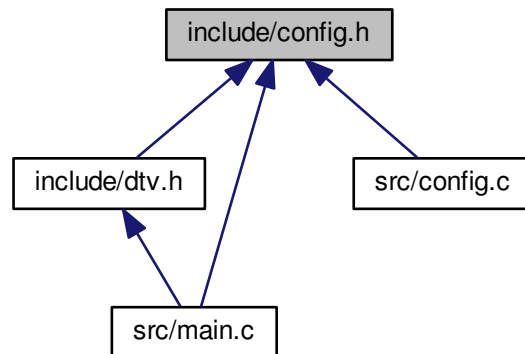
6.2 include/config.h File Reference

Contains configuration file API.

```
#include <stdio.h>
#include "tdp_api.h"
Include dependency graph for config.h:
```



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



Data Structures

- struct [config_init_ch_info](#)

A structure that holds the initial dtv settings.

Functions

- struct [config_init_ch_info](#) [config_get_init_ch_info](#) (FILE *f)

Reads the initial settings from the specified file.

6.2.1 Detailed Description

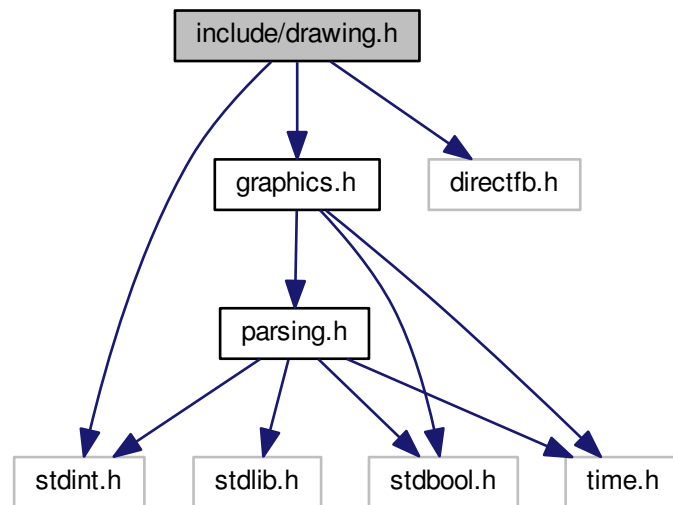
Contains configuration file API.

6.3 include/drawing.h File Reference

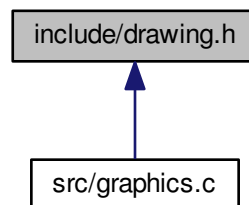
Contains API for drawing graphics elements.

```
#include <stdint.h>
#include <directfb.h>
#include "graphics.h"
```

Include dependency graph for drawing.h:



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



Data Structures

- struct [draw_interface](#)
Basic interface necessary to display graphics elements.

Functions

- `int32_t draw_init (struct draw_interface *draw_i, int *argc, char ***argv)`
Initialize drawing interface.
- `int32_t draw_channel_info (struct draw_interface *draw_i, struct graphics_channel_info info)`

- Draw channel_info graphics element.*
- `int32_t draw_init_message` (struct `draw_interface` *draw_i)
- Draw initializing message.*
- `int32_t draw_time` (struct `draw_interface` *draw_i, struct tm tm)
- Draw time graphics element.*
- `int32_t draw_volume` (struct `draw_interface` *draw_i, uint8_t vol)
- Draw volume graphics element.*
- `int32_t draw_no_channel` (struct `draw_interface` *draw_i)
- Draw no channel display.*
- `int32_t draw_audio_only` (struct `draw_interface` *draw_i)
- Draw radio graphics.*
- `int32_t draw_channel_number` (struct `draw_interface` *draw_i, uint16_t ch_num)
- Draw channel number.*
- `int32_t draw_blackscreen` (struct `draw_interface` *draw_i)
- Draw a black rectangle.*
- `int32_t draw_clear` (struct `draw_interface` *draw_i)
- Clear the screen.*
- `int32_t draw_refresh` (struct `draw_interface` *draw_i)
- Refresh display.*
- `int32_t draw_deinit` (struct `draw_interface` *draw_i)
- Deinitialize drawing interface.*

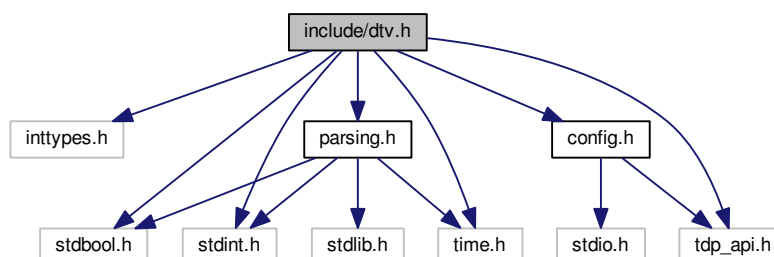
6.3.1 Detailed Description

Contains API for drawing graphics elements.

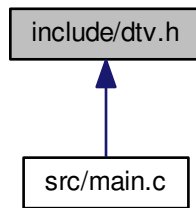
6.4 include/dtv.h File Reference

Contains DTV API.

```
#include <inttypes.h>
#include <stdbool.h>
#include <stdint.h>
#include <time.h>
#include "config.h"
#include "parsing.h"
#include "tdp_api.h"
Include dependency graph for dtv.h:
```



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



Data Structures

- struct [dtv_channel_info](#)
Contains basic channel info.

Functions

- void [dtv_init](#) (struct [config_init_ch_info](#) init_info)
Function that initializes internal DTV state.
- struct [dtv_channel_info](#) [dtv_switch_channel](#) (uint16_t ch_num)
Tries to switch to the desired channel.
- t_Error [dtv_set_volume](#) (uint8_t vol)
Tries to set the volume to the desired value.
- struct tm [dtv_get_time](#) ()
Gets the time information.
- struct [sdt](#) [dtv_get_info](#) (uint16_t ch_num)
Gets the SDT information for the specified channel.
- void [dtv_deinit](#) ()
Deinitializes the internal DTV state.

6.4.1 Detailed Description

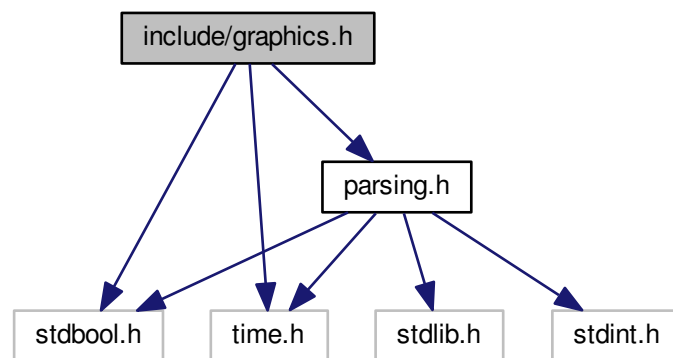
Contains DTV API.

6.5 include/graphics.h File Reference

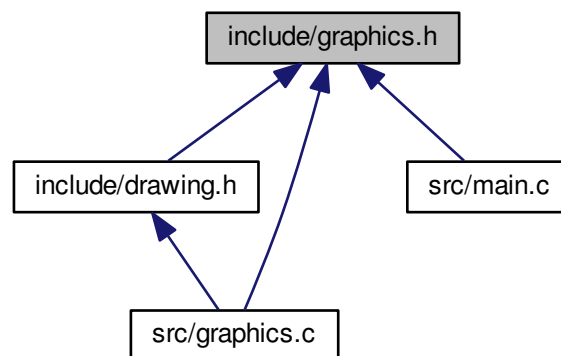
Contains the graphics API.

```
#include <stdbool.h>
#include <time.h>
#include "parsing.h"
```

Include dependency graph for graphics.h:



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



Data Structures

- struct [graphics_channel_info](#)

A struct that contains some basic channel info.

Functions

- void `graphics_show_channel_info` (struct `graphics_channel_info` info)
Displays some basic information about a channel on the screen.
- void `graphics_show_init` ()
Displays initializing message.
- void `graphics_hide_init` ()
Removes initializing message.
- void `graphics_show_time` (struct tm tm)
Displays current time.
- void `graphics_show_volume` (uint8_t vol)
Displays volume information on the screen.
- void `graphics_show_mute` ()
Displays mute symbol.
- void `graphics_hide_mute` ()
Removes mute symbol.
- void `graphics_show_channel_number` (uint16_t ch_num)
Displays selected channel number.
- void `graphics_blackscreen` ()
Puts a black screen on the screen.
- void `graphics_clear` ()
Clears all graphics elements from screen.
- void `graphics_start_render` (int *argc, char ***argv)
Starts rendering graphic elements on screen.
- void `graphics_stop` ()
Stops graphics rendering loop.

6.5.1 Detailed Description

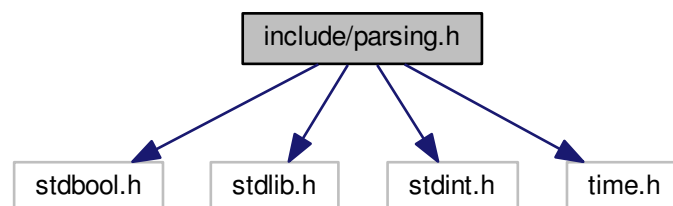
Contains the graphics API.

6.6 include/parsing.h File Reference

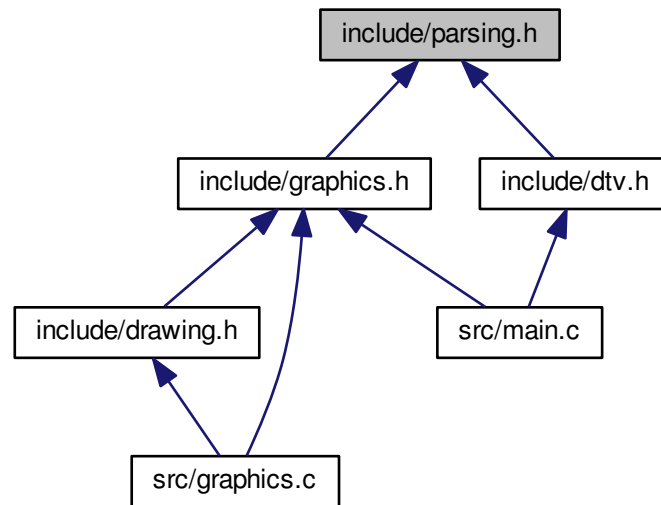
Contains streamlined internal representations of tables.

```
#include <stdbool.h>
#include <stdlib.h>
#include <stdint.h>
#include <time.h>
```

Include dependency graph for parsing.h:



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



Data Structures

- struct [pat](#)
Contains important info from the PAT table.
- struct [pat::pmt_basic](#)
Contains enough info to identify a PMT table.
- struct [pmt](#)
Contains important info from the PMT table.
- struct [sdt](#)
Contains important info from the SDT table.

Functions

- struct [pat](#) [parse_pat](#) (const uint8_t *buffer)
Parses the given buffer as a PAT table.
- struct [pmt](#) [parse_pmt](#) (const uint8_t *buffer)
Parses the given buffer as a PMT table.
- struct tm [parse_tot](#) (const uint8_t *buffer)
Parses the given buffer as a TOT table and converts its info to C representation of time.
- struct [sdt](#) [parse_sdt](#) (const uint8_t *buffer, uint16_t ch_num)
Parses the given buffer as a SDT table and extracts information about the specified channel.

6.6.1 Detailed Description

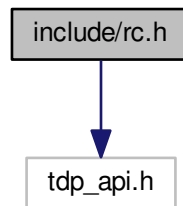
Contains streamlined internal representations of tables.

6.7 include/rc.h File Reference

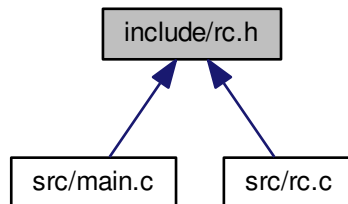
Contains Remote Control API.

```
#include "tdp_api.h"
```

Include dependency graph for rc.h:



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



Macros

- `#define KEY_BACK 1`
Specifies the code of the back key.
- `#define KEY_1 2`
Specifies the code of the 1 key.
- `#define KEY_0 11`
Specifies the code of the 0 key.
- `#define KEY_MUTE 60`
Specifies the code of the mute key.
- `#define KEY_CHANNEL_DOWN 61`
Specifies the code of the channel down key.
- `#define KEY_CHANNEL_UP 62`
Specifies the code of the channelup key.

- `#define KEY_VOLUME_UP 63`
Specifies the code of the volume up key.
- `#define KEY_VOLUME_DOWN 64`
Specifies the code of the volume down key.
- `#define KEY_INFO 358`
Specifies the code of the info key.

Typedefs

- `typedef void(* rc_key_callback) (int key_no)`
A callback that should take action on key press.

Functions

- `void rc_start_loop (const char *dev, rc_key_callback callback)`
A function that starts the loop that waits for input events from the remote control.
- `void rc_stop_loop ()`
Stops the event loop.

6.7.1 Detailed Description

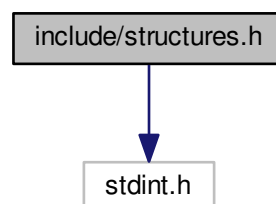
Contains Remote Control API.

6.8 include/structures.h File Reference

Contains nitty-gritty DVB details.

```
#include <stdint.h>
```

Include dependency graph for structures.h:



Data Structures

- struct [table_header](#)
Header that is part of all tables.
- struct [pat_header](#)
Represents PAT header.
- struct [pat_body](#)
Represents PAT body.
- struct [pmt_header](#)
Represents PMT header.
- struct [pmt_body](#)
Represents PMT body.
- struct [teletext_descriptor_header](#)
Represents the teletext descriptor header.
- struct [sdt_header](#)
Represents SDT header.
- struct [sdt_body](#)
Represents SDT body.
- struct [sdt_descriptor1](#)
Represents the first half of the service descriptor.
- struct [sdt_descriptor2](#)
Represents the second half of the service descriptor.
- struct [tot_header](#)
Represents TOT header.
- struct [tot_descriptor_header](#)
Represents TOT descriptor header.
- struct [tot_descriptor_body](#)
Represents TOT descriptor body.

Functions

- struct [table_header](#) **__attribute__((packed))**
- struct [pat_header](#) [get_pat_header](#) (const uint8_t *buffer)
Retrieves [pat_header](#) from the stream.
- struct [pat_body](#) [get_pat_body](#) (const uint8_t *buffer)
Retrieves [pat_body](#) from the stream.
- struct [pmt_header](#) [get_pmt_header](#) (const uint8_t *buffer)
Retrieves [pmt_header](#) from the stream.
- struct [pmt_body](#) [get_pmt_body](#) (const uint8_t *buffer)
Retrieves [pmt_body](#) from the stream.
- struct [teletext_descriptor_header](#) [get_teletext_descriptor_header](#) (const uint8_t *buffer)
Retrieves [teletext_descriptor_header](#) from the stream.
- struct [sdt_header](#) [get_sdt_header](#) (const uint8_t *buffer)
Retrieves [sdt_header](#) from the stream.
- struct [sdt_body](#) [get_sdt_body](#) (const uint8_t *buffer)
Retrieves [sdt_body](#) from the stream.
- struct [sdt_descriptor1](#) [get_sdt_descriptor1](#) (const uint8_t *buffer)
Retrieves [sdt_descriptor1](#) from the stream.
- struct [sdt_descriptor2](#) [get_sdt_descriptor2](#) (const uint8_t *buffer)
Retrieves [sdt_descriptor2](#) from the stream.

- struct [tot_header](#) [get_tot_header](#) (const uint8_t *buffer)
Retrieves [tot_header](#) from the stream.
- struct [tot_descriptor_header](#) [get_tot_descriptor_header](#) (const uint8_t *buffer)
Retrieves [tot_descriptor_header](#) from the stream.
- struct [tot_descriptor_body](#) [get_tot_descriptor_body](#) (const uint8_t *buffer)
Retrieves [tot_descriptor_body](#) from the stream.

Variables

- uint8_t [tid](#)
Table id.
- union {
 struct {
 uint16_t [len](#): 12
 Length of the rest of the table.
 uint16_t [res](#): 2
 uint16_t [zero](#): 1
 uint16_t [ssi](#): 1
 } [b1s](#)
 uint16_t [bitfield1](#)
} [b1u](#)
- struct [table_header](#) [hdr](#)
- uint16_t [tsi](#)
- uint8_t [sec](#)
- uint8_t [lsn](#)
- uint16_t [ch_num](#)
Channel number of the current PMT.
- struct {
 uint8_t [cni](#): 1
 uint8_t [version](#): 5
 uint8_t [res](#): 2
} [b](#)
- union {
 struct {
 uint16_t [pilen](#): 12
 Length of the [pmt_body](#) section.
 uint16_t [res3](#): 4
 } [b2s](#)
 uint16_t [bitfield2](#)
} [b2u](#)
- uint8_t [type](#)
Type of stream.
- struct [teletext_descriptor_header](#) [__attribute__](#)
- uint16_t [oni](#)
- uint16_t [sid](#)
Service id (same as channel number).
- uint8_t [tag](#)
- uint8_t [spnlen](#)
Length of the service provider name.
- uint8_t [snlen](#)
Length of the service name.

- uint8_t [time](#) [5]
Brain-dead encoded time information.
- uint16_t [lto](#)
Local time offset.
- uint8_t [toc](#) [5]
- uint16_t [nto](#)

6.8.1 Detailed Description

Contains nitty-gritty DVB details.

6.8.2 Variable Documentation

6.8.2.1 uint16_t ch_num

Channel number of the current PMT.

Channel number.

6.8.2.2 uint8_t len

Length of the rest of the table.

Length of the descriptor body.

Length of the descriptor.

6.8.2.3 uint16_t pid

PID of the PMT table.

PID of the PS.

6.8.2.4 uint8_t type

Type of stream.

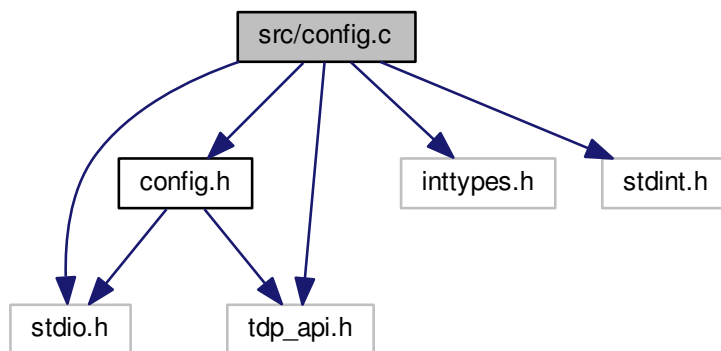
Type of service (channel).

6.9 src/config.c File Reference

Implementation for the configuration file interface.

```
#include "config.h"
#include <inttypes.h>
#include <stdio.h>
#include <stdint.h>
#include "tdp_api.h"
```

Include dependency graph for config.c:



Macros

- `#define BUF_SIZE 100`
- `#define MAKE_GETTER(TYPE, NAME, NOT_FOUND, CONVERSION)`
Creates a getter function for a field. The generated function is named `get_NAME`, and searches through the file, for the field `NAME`, with type `TYPE`, and input conversion `CONVERSION`.

Functions

- `MAKE_GETTER (uint32_t, frequency, NO_FREQUENCY, "%SCNu32")`
Attempts to load frequency from file.
- `MAKE_GETTER (uint32_t, bandwidth, NO_BANDWIDTH, "%SCNu32")`
Attempts to load bandwidth from file.
- `MAKE_GETTER (uint16_t, video_pid, NO_VIDEO_PID, "%SCNu16")`
Attempts to load video PID from file.
- `MAKE_GETTER (uint16_t, audio_pid, NO_AUDIO_PID, "%SCNu16")`
Attempts to load audio PID from file.
- `MAKE_GETTER (uint16_t, ch_num, NO_CH_NUM, "%SCNu16")`
Attempts to load channel number from file.
- `MAKE_GETTER (int, module, NO_MODULE, "%d")`
Attempts to load module type from file.
- `MAKE_GETTER (int, video_type, NO_VIDEO_TYPE, "%d")`

- Attempts to load video type from file.
- `MAKE_GETTER` (int, audio_type, NO_AUDIO_TYPE, "%d")
Attempts to load audio type from file.
- `MAKE_GETTER` (int, teletext, NO_TELETEXT, "%d")
Attempts to load teletext from file.
- struct `config_init_ch_info` `config_get_init_ch_info` (FILE *f)
Attempts to load all fields from the file.

6.9.1 Detailed Description

Implementation for the configuration file interface.

6.9.2 Macro Definition Documentation

6.9.2.1 #define BUF_SIZE 100

Parameters

<i>Buffer</i>	size for reading the file.
---------------	----------------------------

6.9.2.2 #define MAKE_GETTER(TYPE, NAME, NOT_FOUND, CONVERSION)

Value:

```
static TYPE get_#NAME(FILE *f) \
{ \
    rewind(f); \
    while (!feof(f)) \
    { \
        char buf[BUF_SIZE]; \
        fgets(buf, BUF_SIZE, f); \
        \
        TYPE ret; \
        if (sscanf(buf, #NAME" = "CONVERSION, &ret) == 1) \
            return ret; \
    } \
    \
    return NOT_FOUND; \
}
```

Creates a getter function for a field. The generated function is named `get_NAME`, and searches through the file, for the field `NAME`, with type `TYPE`, and input conversion `CONVERSION`.

Parameters

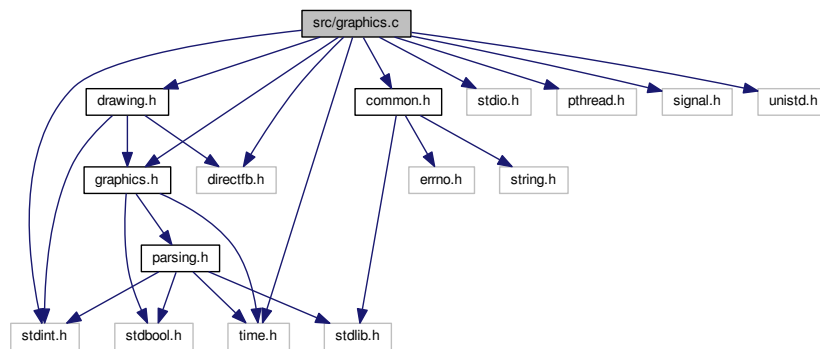
<i>TYPE</i>	type of the field, and also return type of the function.
<i>NAME</i>	name of the field, which is also part of the function name.
<i>NOT_FOUND</i>	value of type <code>TYPE</code> , to be returned if field was not found.
<i>CONVERSION</i>	A compile-time string that specifies the input conversion à la <code>scanf</code> .

6.10 src/graphics.c File Reference

Contains implementation for graphics interface.

```
#include <stdint.h>
#include <stdio.h>
#include <time.h>
#include <pthread.h>
#include <signal.h>
#include <unistd.h>
#include <directfb.h>
#include "common.h"
#include "drawing.h"
#include "graphics.h"
#include "graphics.h"
```

Include dependency graph for graphics.c:



Data Structures

- struct [graphics_flags](#)
A struct that keeps the state of what should be displayed.
- struct [graphics_args](#)
A shim structure to pass main arguments to DirectFB.

Macros

- `#define _POSIX_C_SOURCE 200809L`
- `#define LOG_GRAPHICS(fmt, ...) LOG("Graphics", fmt, ##__VA_ARGS__)`
- `#define DRAWCHECK(err)`

Enumerations

- enum [g_error](#) { [G_ERROR](#) = -1, [G_NO_ERROR](#) }
Error codes to be used for internal graphics functions.

Functions

- void [graphics_show_channel_info](#) (struct [graphics_channel_info](#) info)
Displays some basic information about a channel on the screen.
- void [graphics_show_time](#) (struct tm tm)
Displays current time.
- void [graphics_show_volume](#) (uint8_t vol)
Displays volume information on the screen.
- void [graphics_show_init](#) ()
Displays initializing message.
- void [graphics_hide_init](#) ()
Removes initializing message.
- void [graphics_show_mute](#) ()
Displays mute symbol.
- void [graphics_hide_mute](#) ()
Removes mute symbol.
- void [graphics_show_channel_number](#) (uint16_t ch_num)
Displays selected channel number.
- void [graphics_blackscreen](#) ()
Puts a black screen on the screen.
- void [graphics_clear](#) ()
Clears all graphics elements from screen.
- enum [g_error](#) [graphics_render](#) (int *argc, char ***argv)
Function that continuously refreshes graphics display according to the current state of [graphics_flags](#).
- void [graphics_start_render](#) (int *argc, char ***argv)
Starts rendering graphic elements on screen.
- void [graphics_stop](#) ()
Stops graphics rendering loop.

6.10.1 Detailed Description

Contains implementation for graphics interface.

6.10.2 Macro Definition Documentation

6.10.2.1 #define DRAWCHECK(err)

Value:

```
{ \
    if (err != EXIT_SUCCESS) \
    { \
        fprintf(stderr, "%s:%d:%s\n", __FILE__, __LINE__, __func__); \
        release(); \
        return G\_ERROR; \
    } \
}
```

6.10.3 Enumeration Type Documentation

6.10.3.1 enum g_error

Error codes to be used for internal graphics functions.

Enumerator

G_ERROR Specifies that a graphics error has occurred.

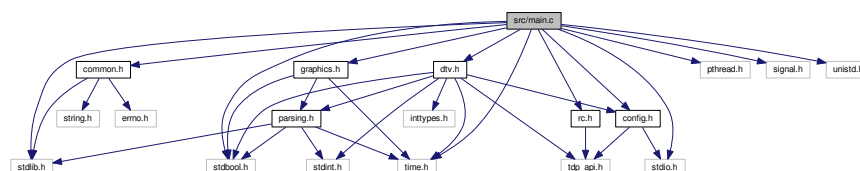
G_NO_ERROR Specifies that a graphics_error has *not* occurred.

6.11 src/main.c File Reference

Contains the implementation that glues other modules together.

```
#include <stdbool.h>
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <pthread.h>
#include <signal.h>
#include <unistd.h>
#include "common.h"
#include "graphics.h"
#include "config.h"
#include "dtv.h"
#include "rc.h"
```

Include dependency graph for main.c:



Macros

- `#define _POSIX_C_SOURCE 200809L`
- `#define LOG_MAIN(fmt, ...) LOG("Main", fmt, ##__VA_ARGS__)`

Functions

- void `handle_signal` (int signum)
Function that handles SIGINT and SIGTERT, by ensuring graceful exit.
- void `calculate_time` ()
Function that calculates the amount of time that has passed since TOT table was received, and adjusts the time to be displayed accordingly.
- void `confirm_channel` (union sigval s)
Function that switches channels once the change has been confirmed.
- void `react_to_keypress` (int key_code)
Function that reacts to a keypress from the remote control.
- int `main` (int argc, char **argv)

6.11.1 Detailed Description

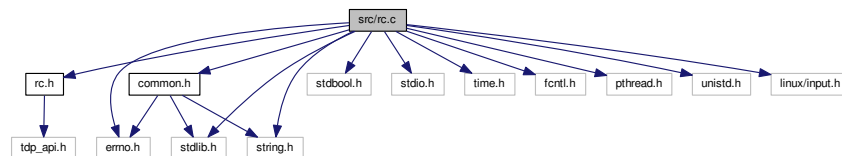
Contains the implementation that glues other modules together.

6.12 src/rc.c File Reference

Contains the implementation of the remote control interface.

```
#include "rc.h"
#include <errno.h>
#include <stdbool.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <time.h>
#include <fcntl.h>
#include <pthread.h>
#include <unistd.h>
#include <linux/input.h>
#include "common.h"
```

Include dependency graph for rc.c:



Data Structures

- struct [rc_args](#)
A shim structure to pass arguments to event thread.

Macros

- `#define _POSIX_C_SOURCE 200809L`
- `#define LOG_RC(fmt, ...) LOG("RC", fmt, ##__VA_ARGS__)`
- `#define EVENT_KEY_PRESS 1`
Specifies that a key press has occurred.

Functions

- void [rc_start_loop](#) (const char *dev, [rc_key_callback](#) kc)
A function that starts the loop that waits for input events from the remote control.
- void [rc_stop_loop](#) ()
Stops the event loop.

6.12.1 Detailed Description

Contains the implementation of the remote control interface.

Index

- BUF_SIZE
 - config.c, [49](#)
- ch_num
 - structures.h, [47](#)
- common.h
 - FAIL_STD, [34](#)
 - FAIL, [34](#)
 - LOG, [34](#)
- config.c
 - BUF_SIZE, [49](#)
 - MAKE_GETTER, [49](#)
- config_get_init_ch_info
 - Configuration file interface, [7](#)
- config_init_ch_info, [17](#)
- Configuration file interface, [7](#)
 - config_get_init_ch_info, [7](#)
- DRAWCHECK
 - graphics.c, [51](#)
- DTV interface, [9](#)
 - dtv_set_volume, [9](#)
- DVB table retrieval interface, [14](#)
- draw_interface, [18](#)
- Drawing interface, [8](#)
- dtv_channel_info, [18](#)
- dtv_set_volume
 - DTV interface, [9](#)
- FAIL_STD
 - common.h, [34](#)
- FAIL
 - common.h, [34](#)
- G_ERROR
 - graphics.c, [52](#)
- G_NO_ERROR
 - graphics.c, [52](#)
- g_error
 - graphics.c, [52](#)
- Graphics interface, [10](#)
 - graphics_show_volume, [10](#)
- graphics.c
 - DRAWCHECK, [51](#)
 - G_ERROR, [52](#)
 - G_NO_ERROR, [52](#)
 - g_error, [52](#)
- graphics_args, [19](#)
- graphics_channel_info, [19](#)
- graphics_flags, [20](#)
- graphics_show_volume
 - Graphics interface, [10](#)
- include/common.h, [33](#)
- include/config.h, [35](#)
- include/drawing.h, [36](#)
- include/dtv.h, [38](#)
- include/graphics.h, [40](#)
- include/parsing.h, [41](#)
- include/rc.h, [43](#)
- include/structures.h, [44](#)
- LOG
 - common.h, [34](#)
- len
 - structures.h, [47](#)
- MAKE_GETTER
 - config.c, [49](#)
- pat, [21](#)
- pat::pmt_basic, [24](#)
- pat_body, [22](#)
- pat_header, [22](#)
- pid
 - structures.h, [47](#)
- pmt, [23](#)
- pmt_body, [24](#)
- pmt_header, [25](#)
- rc_args, [26](#)
- rc_start_loop
 - Remote-control interface, [13](#)
- Remote-control interface, [13](#)
 - rc_start_loop, [13](#)
- sdt, [26](#)
- sdt_body, [27](#)
- sdt_descriptor1, [27](#)
- sdt_descriptor2, [28](#)
- sdt_header, [29](#)
- src/config.c, [48](#)
- src/graphics.c, [50](#)
- src/main.c, [52](#)
- src/rc.c, [53](#)
- structures.h
 - ch_num, [47](#)
 - len, [47](#)
 - pid, [47](#)
 - type, [47](#)

Table parsing interface, [12](#)
table_header, [29](#)
teletext_descriptor_header, [30](#)
tot_descriptor_body, [31](#)
tot_descriptor_header, [31](#)
tot_header, [32](#)
type
 structures.h, [47](#)