libfg Reference Manual

1.2

Generated by Doxygen 1.3.4

Thu Nov 20 01:38:21 2003

Contents

Chapter 1

libfg - Framegrabber Library

1.1 Introduction

1.2 Capturing Video

Chapter 2

libfg Data Structure Index

2.1 libfg Data Structures

Here are the data structures with brief descriptions:

Chapter 3

libfg File Index

3.1 libfg File List

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

Chapter 4

libfg Data Structure Documentation

4.1 FRAME Struct Reference

A single frame buffer.

`#include <frame.h>`

Data Fields

- int **width**

  width in pixels

- int **height**

  height in pixels

- int **depth**

  bit depth (bits per pixel)

- int **format**

  VIDEO_∗ format.

- void ∗ **data**

  pointer to data buffer

4.1.1 Detailed Description

A single frame buffer.

Represents a single image in the output from the frame grabber. Carries with it the dimensions, format and the data buffer. The type of the data depends on the format flag (uses the VIDEO_∗ flags from Video4Linux), so RGB24 would be a triplet of chars, while RGB32 would be an int.

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

- **frame.h**

4.2 FRAME_RGB Struct Reference

```
#include <frame.h>
```

Data Fields

- char **red**
- char **green**
- char **blue**

4.2.1 Detailed Description

A 24-bit RGB component pixel

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

- **frame.h**

4.3 FRAMEGRABBER Struct Reference

An opaque framegrabber handle.

`#include <capture.h>`

Data Fields

- char * **device**

  Device name, eg. "/dev/video".

- int **fd**

  File handle for open device.

- video_capability **caps**

  Capabilities.

- video_channel * **sources**

  Input sources (eg. TV, SVideo).

- int **source**

  Currently selected source.

- video_tuner **tuner**

  TV or Radio tuner.

- video_window **window**

  Capture window.

- video_picture **picture**

  Picture controls (eg. bright).

- video_mmap **mmap**

  Memory-mapped info.

- video_buffer **fbuffer**

  Frame buffer.

- video_mbuf **mbuf**

  Memory buffer #frames, offsets.

- void * **mb_map**

  Memory-mapped buffer.

- int **cur_frame**

  Currently capuring frame no.

4.3.1 Detailed Description

An opaque framegrabber handle.

Represents all information about a frame grabber device. Returned by **fg_open()**(p. 14), and used as the first parameter for all other fg_∗() calls.

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

- **capture.h**

Chapter 5

libfg File Documentation

5.1 capture.h File Reference

Capture client interface.

`#include <stdio.h>`

`#include <fcntl.h>`

`#include <stdlib.h>`

`#include <unistd.h>`

`#include <sys/mman.h>`

`#include <sys/ioctl.h>`

`#include <linux/fs.h>`

`#include <linux/kernel.h>`

`#include <linux/videodev.h>`

`#include "frame.h"`

Data Structures

- struct **FRAMEGRABBER**

  An opaque framegrabber handle.

Defines

- #define **FG_DEFAULT_DEVICE** "/dev/video" /** Default video input */
- #define **FG_DEFAULT_WIDTH** 640
- #define **FG_DEFAULT_HEIGHT** 480
- #define **FG_PERCENT**(n) ((n)*65535/100)
- #define **FG_50PC** FG_PERCENT(50)
- #define **FG_SOURCE_TV** 0
- #define **FG_SOURCE_COMPOSITE** 1
- #define **FG_SOURCE_SVIDEO** 2

Functions

- **FRAMEGRABBER** ∗ **fg_open** (const char ∗dev)
- void **fg_close** (**FRAMEGRABBER** ∗fg)
- **FRAME** ∗ **fg_grab** (**FRAMEGRABBER** ∗fg)
- **FRAME** ∗ **fg_grab_frame** (**FRAMEGRABBER** ∗fg, **FRAME** ∗fr)
- int **fg_set_source** (**FRAMEGRABBER** ∗fg, int src)
- int **fg_set_source_norm** (**FRAMEGRABBER** ∗fg, int norm)
- int **fg_get_source_count** (**FRAMEGRABBER** ∗fg)
- char ∗ **fg_get_source_name** (**FRAMEGRABBER** ∗fg, int src)
- int **fg_set_channel** (**FRAMEGRABBER** ∗fg, float freq)
- float **fg_get_channel** (**FRAMEGRABBER** ∗fg)
- int **fg_set_format** (**FRAMEGRABBER** ∗fg, int fmt)
- int **fg_set_capture_window** (**FRAMEGRABBER** ∗fg, int x, int y, int width, int height)
- int **fg_set_brightness** (**FRAMEGRABBER** ∗fg, int br)
- int **fg_set_hue** (**FRAMEGRABBER** ∗fg, int hu)
- int **fg_set_colour** (**FRAMEGRABBER** ∗fg, int co)
- int **fg_set_color** (**FRAMEGRABBER** ∗fg, int co)
- int **fg_set_contrast** (**FRAMEGRABBER** ∗fg, int ct)
- int **fg_set_whiteness** (**FRAMEGRABBER** ∗fg, int wh)
- **FRAME** ∗ **fg_new_compatible_frame** (**FRAMEGRABBER** ∗fg)
- void **fg_dump_info** (**FRAMEGRABBER** ∗fg)

5.1.1 Detailed Description

Capture client interface.

libfg - Frame Grabber interface for Linux

Provides a high-level C interface for controlling frame grabber and TV tuner cards. Uses the Video 4 Linux API (currently v1) and thus supports any V4L supported device.

Author:
Gavin Baker <`gavinb@antonym.org`>

VersRevision
1.9

Homepage: `http://www.antonym.org/libfg`

5.1.2 Function Documentation

5.1.2.1 void fg_close (FRAMEGRABBER ∗ *fg*)

Closes an open framegrabber device, and releases all memory allocated with it.

Parameters:
fg The framegrabber handle to close.

5.1.2.2 void fg_dump_info (FRAMEGRABBER * *fg*)

Dumps to the console on stdout all the status information available for the framegrabber.

Parameters:

fg Framegrabber handle

5.1.2.3 float fg_get_channel (FRAMEGRABBER * *fg*)

Queries the current frequency of the TV tuner.

Parameters:

fg Framegrabber handle

Returns:

The frequency in MHz

5.1.2.4 int fg_get_source_count (FRAMEGRABBER * *fg*)

Returns the number of input sources available.

Parameters:

fg Framegrabber handle

Returns:

>0 Sources (can be used in fg_set_source)

5.1.2.5 char* fg_get_source_name (FRAMEGRABBER * *fg*, int *src*)

Returns a user-friendly name corresponding to the supplied channel number.

Parameters:

fg Framegrabber handle

src Source id (eg. FG_SOURCE_TV)

Returns:

Name, like "Television"

5.1.2.6 FRAME* fg_grab (FRAMEGRABBER * *fg*)

Reads a frame from the capture device, allocating a new **FRAME**(p. 7) instance and returning it. The frame will be allocated the maximum size window /// in the default picture format. Note that this is a *blocking* read, /// and thus will wait until the next frame is ready. The caller is /// responsible for doing a **frame_release()**(p. 19) when done with the frame (to /// free memory).

Parameters:

fg The framegrabber handle from which to capture

Returns:
The most recently captured frame, or NULL on error

Note:
This function blocks!

5.1.2.7 FRAME* fg_grab_frame (FRAMEGRABBER * *fg*, FRAME * *fr*)

Reads a frame from the capture device, using the existing frame storage as passed in. Returns the same instance, with the contents of /// the last frame. Note that this is a *blocking* read, and thus will /// wait until the next frame is ready.

Parameters:
fg The open framegrabber
fr An existing frame

Returns:
The most recently captured frame, or NULL on error

Note:
This function blocks! The size *must* be correct!

5.1.2.8 FRAME* fg_new_compatible_frame (FRAMEGRABBER * *fg*)

Returns a newly allocated frame that is compatible with the current frame grabber settings; that is, the window width and height, and the capture format. This frame must be deleted by the caller with **frame_release()**(p. 19).

Returns:
A new frame

5.1.2.9 FRAMEGRABBER* fg_open (const char * *dev*)

Opens and initialises the frame grabber device with some reasonable default values, and queries for all capabilities.

Parameters:
dev Device name to open, eg. "/dev/video2" or NULL for "/dev/video".

Returns:
The open framegrabber handle, or NULL in the case of an error.

5.1.2.10 int fg_set_brightness (FRAMEGRABBER * *fg*, int *br*)

Sets the picture brightness to the specified value.

Parameters:
fg Framegrabber handle

br Brightness (in percent)

Returns:

0 Success -1 Failure

5.1.2.11 int fg_set_capture_window (FRAMEGRABBER * *fg*, int *x*, int *y*, int *width*, int *height*)

Specifies a sub-window of the input source to capture. The parameters specify the capture window that is smaller than or equal to the maximum supported window size.

Parameters:

fg Framegrabber handle

x

y

width

height

Returns:

0 Success -1 Failure

5.1.2.12 int fg_set_channel (FRAMEGRABBER * *fg*, float *freq*)

Sets the TV tuner to the specified frequency.

Parameters:

fg Framegrabber handle

freq Tuner frequency, in MHz

Returns:

0 Success, tuned in -1 Failure

5.1.2.13 int fg_set_color (FRAMEGRABBER * *fg*, int *co*)

Sets the picture color balance for Americans to the specified value.

Parameters:

fg Framegrabber handle

co Color balance (in percent)

Returns:

0 Success -1 Failure

5.1.2.14 int fg_set_colour (FRAMEGRABBER * *fg*, int *co*)

Sets the picture colour balance for Queen's English speakers to the specified value.

Parameters:

fg Framegrabber handle

co Colour balance (in percent)

Returns:

0 Success -1 Failure

5.1.2.15 int fg_set_contrast (FRAMEGRABBER * *fg*, int *ct*)

Sets the picture contrast to the specified value.

Parameters:

fg Framegrabber handle

ct Contrast (in percent)

Returns:

0 Success -1 Failure

5.1.2.16 int fg_set_format (FRAMEGRABBER * *fg*, int *fmt*)

Specifies the capture format to use. Must be one of the VIDEO_PALETTE_* flags.

Parameters:

fg Framegrabber handle

fmt pixel format

Note:

Currently only RGB32 and RGB24 are properly supported.

Returns:

0 Success

5.1.2.17 int fg_set_hue (FRAMEGRABBER * *fg*, int *hu*)

Sets the picture hue control to the specified value.

Parameters:

fg Framegrabber handle

hu Hue (in percent)

Returns:

0 Success -1 Failure

5.1.2.18 int fg_set_source (FRAMEGRABBER ∗ *fg*, int *src*)

Specifies the number of the video source to be used for the input signal. For example, tuner, composite or S/Video signal.

Parameters:

fg Framegrabber handle

src Source id (eg. FG_SOURCE_SVIDEO)

Returns: 0 on success, -1 on failure

5.1.2.19 int fg_set_source_norm (FRAMEGRABBER ∗ *fg*, int *norm*)

Specifies the video signal norm (eg. PAL, NTSC, SECAM) for the current input source.

Parameters:

fg Framegrabber handle

norm Signal norm (eg. VIDEO_MODE_PAL)

Returns:

0 On success -1 Failure

5.1.2.20 int fg_set_whiteness (FRAMEGRABBER ∗ *fg*, int *wh*)

Sets the picture white balance to the specified value.

Parameters:

fg Framegrabber handle

wh Whiteness (in percent)

Returns:

0 Success -1 Failure

5.2 frame.h File Reference

Frame interface.

Data Structures

- struct **FRAME**

  A single frame buffer.

- struct **FRAME_RGB**

Functions

- **FRAME** * **frame_new** (int width, int height, int format)

  Create a new frame.

- void **frame_release** (**FRAME** *fr)
- void * **frame_get_data** (**FRAME** *fr)
- int **frame_get_size** (**FRAME** *fr)
- int **frame_get_width** (**FRAME** *fr)
- int **frame_get_height** (**FRAME** *fr)
- int **frame_save** (**FRAME** *fr, const char *filename)

5.2.1 Detailed Description

Frame interface.

libfg - Frame Grabber interface for Linux

Each frame captured by the **FRAMEGRABBER**(p. 9) returns a **FRAME**(p. 7) (defined here). It contains the raw frame data, as well as information about the frame's size and format.

Author:

Gavin Baker <gavinb@antonym.org>

VersRevision

1.4

5.2.2 Function Documentation

5.2.2.1 void* frame_get_data (FRAME * *fr*)

Returns a pointer to the raw frame data.

Parameters:

fr The frame

5.2.2.2 int frame_get_height (FRAME * *fr*)

Returns the size of the frame, given the dimensions and the pixel format.

Parameters:

fr The frame

5.2.2.3 int frame_get_size (FRAME * *fr*)

Returns the size of the frame, given the dimensions and the pixel format.

Parameters:

fr The frame

5.2.2.4 int frame_get_width (FRAME * *fr*)

Returns the size of the frame, given the dimensions and the pixel format.

Parameters:

fr The frame

5.2.2.5 FRAME* frame_new (int *width*, int *height*, int *format*)

Create a new frame.

Creates a new frame buffer, of the given dimensions, for the specified pixel format.

Parameters:

width Width to allocate (pixels)

height Height to allocate (pixels)

format Pixel format (VIDEO_* flags)

Returns:

A new allocated frame buffer

5.2.2.6 void frame_release (FRAME * *fr*)

Releases a frame and all its associated memory.

Parameters:

fr The frame to release

5.2.2.7 int frame_save (FRAME * *fr*, const char * *filename*)

Saves the frame to a PNM file for external viewing

Parameters:

fr The frame to save

filename The output filename (eg. "capture.pnm")

Index