

APO - Scanner

Generated by Doxygen 1.9.4

| | |
|---|----------|
| 1 README | 1 |
| 1.1 Introduction | 1 |
| 1.1.1 Scanning Documents from Photographs Using C and MicroZed Camera | 1 |
| 1.2 Dependencies | 1 |
| 1.2.1 LGSL | 1 |
| 1.2.2 LJPEG | 1 |
| 1.3 Project Links | 1 |
| 1.4 Installation Manual | 2 |
| 1.5 User Manual | 2 |
| 1.5.1 Running the Application | 2 |
| 1.6 Conclusion | 2 |
| 1.7 License | 3 |
| 2 File Index | 5 |
| 2.1 File List | 5 |
| 3 File Documentation | 7 |
| 3.1 diod.c File Reference | 7 |
| 3.1.1 Detailed Description | 8 |
| 3.1.2 Function Documentation | 8 |
| 3.1.2.1 diod_set_color() | 8 |
| 3.2 diod.h File Reference | 8 |
| 3.2.1 Detailed Description | 9 |
| 3.2.2 Function Documentation | 9 |
| 3.2.2.1 diod_set_color() | 10 |
| 3.3 diod.h | 10 |
| 3.4 font_types.h | 10 |
| 3.5 img_reader.c File Reference | 11 |
| 3.5.1 Detailed Description | 11 |
| 3.5.2 Function Documentation | 12 |
| 3.5.2.1 read_image() | 12 |
| 3.5.2.2 save_image() | 12 |
| 3.6 img_reader.h File Reference | 12 |
| 3.6.1 Detailed Description | 13 |
| 3.6.2 Function Documentation | 13 |
| 3.6.2.1 read_image() | 14 |
| 3.6.2.2 save_image() | 14 |
| 3.7 img_reader.h | 14 |
| 3.8 knob.c File Reference | 15 |
| 3.8.1 Detailed Description | 15 |
| 3.8.2 Function Documentation | 15 |
| 3.8.2.1 get_knob_value() | 15 |
| 3.9 knob.h File Reference | 16 |

| | |
|--|----|
| 3.9.1 Detailed Description | 17 |
| 3.9.2 Function Documentation | 17 |
| 3.9.2.1 get_knob_value() | 17 |
| 3.10 knob.h | 17 |
| 3.11 lcd.c File Reference | 17 |
| 3.11.1 Detailed Description | 18 |
| 3.11.2 Function Documentation | 19 |
| 3.11.2.1 lcd_char_width() | 19 |
| 3.11.2.2 lcd_color() | 19 |
| 3.11.2.3 lcd_draw_char() | 19 |
| 3.11.2.4 lcd_draw_char_bitmap() | 20 |
| 3.11.2.5 lcd_draw_image() | 20 |
| 3.11.2.6 lcd_draw_pixel() | 21 |
| 3.11.2.7 lcd_draw_plus() | 21 |
| 3.11.2.8 lcd_draw_scaled_pixel_block() | 22 |
| 3.11.2.9 lcd_draw_text() | 22 |
| 3.11.2.10 lcd_fill_screen() | 23 |
| 3.11.2.11 lcd_grey() | 23 |
| 3.11.2.12 lcd_init() | 23 |
| 3.11.2.13 lcd_update_display() | 24 |
| 3.12 main.c File Reference | 24 |
| 3.12.1 Detailed Description | 25 |
| 3.12.2 Function Documentation | 25 |
| 3.12.2.1 capture_points() | 25 |
| 3.12.2.2 initialize_display() | 25 |
| 3.12.2.3 initialize_memory() | 26 |
| 3.12.2.4 load_image() | 26 |
| 3.12.2.5 main() | 27 |
| 3.12.2.6 save_transformed_image() | 27 |
| 3.12.2.7 transform_image() | 27 |
| 3.13 menu.c File Reference | 28 |
| 3.13.1 Detailed Description | 28 |
| 3.13.2 Function Documentation | 28 |
| 3.13.2.1 get_file_name() | 28 |
| 3.14 menu.h File Reference | 29 |
| 3.14.1 Detailed Description | 30 |
| 3.14.2 Function Documentation | 30 |
| 3.14.2.1 get_file_name() | 30 |
| 3.15 menu.h | 30 |
| 3.16 mzap0_parlcd.h | 30 |
| 3.17 mzap0_phys.h | 31 |
| 3.18 mzap0_regs.h | 31 |

| | |
|--|-----------|
| 3.19 scanner.c File Reference | 33 |
| 3.19.1 Detailed Description | 33 |
| 3.19.2 Function Documentation | 33 |
| 3.19.2.1 apply_perspective_transform() | 33 |
| 3.19.2.2 compute_perspective_transform() | 34 |
| 3.19.2.3 print_matrix() | 34 |
| 3.20 scanner.h File Reference | 34 |
| 3.20.1 Detailed Description | 35 |
| 3.20.2 Function Documentation | 35 |
| 3.20.2.1 apply_perspective_transform() | 36 |
| 3.20.2.2 compute_perspective_transform() | 36 |
| 3.20.2.3 print_matrix() | 36 |
| 3.21 scanner.h | 37 |
| 3.22 serialize_lock.h | 37 |
| Index | 39 |

Chapter 1

README

1.1 Introduction

1.1.1 Scanning Documents from Photographs Using C and MicroZed Camera

Author: Anatolii Filkin

Date: 20.04.2024

Project Description: This application is designed to scan documents by capturing a photograph of a textual document and processing it to produce a rectangular output image. It assumes that the input photograph contains an A4 paper. The photograph can be taken using SSH. The user selects four corner points of the document, and the application returns the scanned image. The LED diode displays the intensity of the currently selected pixel.

1.2 Dependencies

1.2.1 LGSL

- **Description:** LGSL (GNU Scientific Library) is a numerical library for C language. It provides a wide range of mathematical routines such as random number generators, special functions, and least-squares fitting.
- **License:** GNU General Public License (GPL)

1.2.2 LJPEG

- **Description:** LJPEG is a JPEG image compression library that allows for reading and writing JPEG images. It is used in this project for handling image input and output.
- **License:** Independent JPEG Group's (IJG) JPEG library license

1.3 Project Links

- ****Git Project Link:****https://github.com/filkiana/APO_SEM

1.4 Installation Manual

1. Clone the Repository:

```
git clone https://github.com/filkiana/APO_SEM
cd APO_SEM
git checkout dev
```

2. Install LGSL and LJPEG Libraries:

```
chmod +x install.sh
./install.sh
```

3. Configure Makefile board ip:

```
vim Makefile
```

Change the IP address in the BOARD_IP variable to the IP address of the MicroZed board.

4. Build the Project:

```
make
```

1.5 User Manual

1.5.1 Running the Application

1. Build and Run the Application on the board using make:

```
make run
```

2. Chose file for scanning using keyboard:

3. Select four corner points of the document using the knobs:

to mark the corners of the document, use the knobs (red for y and blue for x) to move the red-cross cursor and press the green knob to mark the corner.

1. Top left corner
2. Top right corner
3. Bottom right corner
4. Bottom left corner

4. Get the scanned image:

The scanned image will be saved in the same directory as application.

1.6 Conclusion

This project demonstrates the integration of multiple hardware components (LCD display, knobs, RGB DIODA, SSH) and software libraries (LGSL, LJPEG) to create a functional document scanning application on the MicroZed platform. By leveraging user input and real-time image processing, the application provides a practical solution for capturing and processing document images. User gets commands on the display by font texts. The LED feedback system enhances user interaction, making the application intuitive and efficient. Whole project written in C language and have nice modular structure. The code is well-documented and follows the coding style. The project is version-controlled using Git, with meaningful commit messages and a separate development branch.

1.7 License

MIT License

Copyright (c) 2024 Anatolii Filkin

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Chapter 2

File Index

2.1 File List

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

| | | |
|----------------------------------|---|----|
| diod.c | Implementation of Diod LED control functions | 7 |
| diod.h | Header file for LED control functions | 8 |
| font_types.h | | ?? |
| img_reader.c | Image reading and saving functions using JPEG format | 11 |
| img_reader.h | Header file for image reading and saving functions | 12 |
| knob.c | Implementation of knob value reading functions | 15 |
| knob.h | Header file for knob value reading functions | 16 |
| lcd.c | Implementation of LCD display functions | 17 |
| main.c | Main file for the MicroZed based MZ_APO board project | 24 |
| menu.c | Implementation of menu functions | 28 |
| menu.h | Header file for menu functions | 29 |
| mzap0_parlcd.h | | ?? |
| mzap0_phys.h | | ?? |
| mzap0_regs.h | | ?? |
| scanner.c | Implementation of matrix and transformation functions | 33 |
| scanner.h | Header file for matrix and transformation functions | 34 |
| serialize_lock.h | | ?? |

Chapter 3

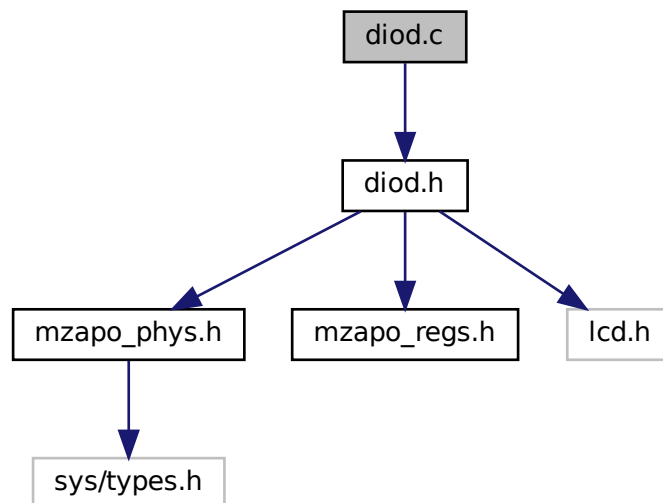
File Documentation

3.1 diod.c File Reference

Implementation of Diod LED control functions.

```
#include "diod.h"
```

Include dependency graph for diod.c:



Functions

- void `diod_set_color` (unsigned char *spiled_base, unsigned char r, unsigned char g, unsigned char b)
Set the color of the Diod.

3.1.1 Detailed Description

Implementation of Diod LED control functions.

3.1.2 Function Documentation

3.1.2.1 diod_set_color()

```
void diod_set_color (
    unsigned char * spiled_base,
    unsigned char r,
    unsigned char g,
    unsigned char b )
```

Set the color of the Diod.

Parameters

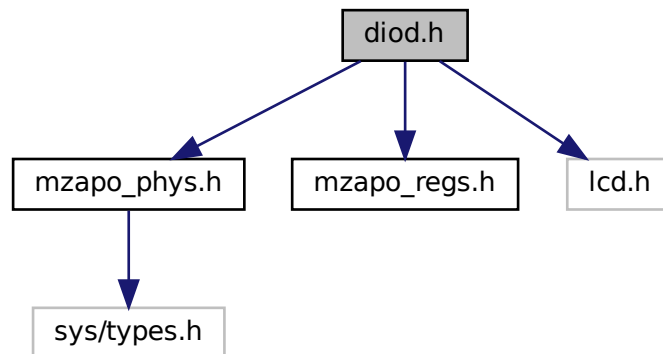
| | |
|--------------------|-------------------------------|
| <i>spiled_base</i> | Base address of the SPILED. |
| <i>r</i> | Red component of the color. |
| <i>g</i> | Green component of the color. |
| <i>b</i> | Blue component of the color. |

3.2 diod.h File Reference

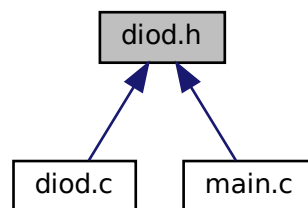
Header file for LED control functions.

```
#include "mzapophys.h"
#include "mzaporegs.h"
#include "lcd.h"
```

Include dependency graph for diod.h:



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



Functions

- void `diod_set_color` (unsigned char *spiled_base, unsigned char r, unsigned char g, unsigned char b)
Set the color of the Diod.

3.2.1 Detailed Description

Header file for LED control functions.

3.2.2 Function Documentation

3.2.2.1 diod_set_color()

```
void diod_set_color (
    unsigned char * spiled_base,
    unsigned char r,
    unsigned char g,
    unsigned char b )
```

Set the color of the Diod.

Parameters

| | |
|--------------------|-------------------------------|
| <i>spiled_base</i> | Base address of the SPILED. |
| <i>r</i> | Red component of the color. |
| <i>g</i> | Green component of the color. |
| <i>b</i> | Blue component of the color. |

3.3 diod.h

[Go to the documentation of this file.](#)

```
1
2
3 #ifndef DIOD_H
4 #define DIOD_H
5 #include "mzapophys.h"
6 #include "mzaporegs.h"
7 #include "lcd.h"
8
9 void diod_set_color(unsigned char *spiled_base, unsigned char r, unsigned char g, unsigned char b);
10
11 #endif // DIOD_H
```

3.4 font_types.h

```
1 /*****
2
3 font_types.h      - simple bitmap fonts type definition
4
5 Simplified font type descriptor based on
6 Microwindows/Nano-X library by Greg Haerr
7
8 https://github.com/ghaerr/microwindows
9
10 Copyright (c) 1999, 2000, 2001, 2002, 2003, 2005, 2010, 2011 Greg Haerr <greg@censoft.com>
11 Portions Copyright (c) 2002 by Koninklijke Philips Electronics N.V.
12
13 Simplification by Pavel Pisa for Czech Technical University
14 Computer Architectures course
15
16 *****/
17
18 #ifndef FONT_TYPES_H
19 #define FONT_TYPES_H
20
21 #include <stdint.h>
22
23 #ifdef __cplusplus
24 extern "C" {
25 #endif
26
27 typedef uint16_t font_bits_t;
28
29 /* builtin C-based proportional/fixed font structure*/
30 typedef struct {
31     char *          name;          /* font name*/
32     int             maxwidth;      /* max width in pixels*/
```



```

33     unsigned int    height;          /* height in pixels*/
34     int            ascent;           /* ascent (baseline) height*/
35     int            firstchar;        /* first character in bitmap*/
36     int            size;             /* font size in characters*/
37     const font_bits_t *bits;         /* 16-bit right-padded bitmap data*/
38     const uint32_t *offset;          /* offsets into bitmap data*/
39     const unsigned char *width;      /* character widths or 0 if fixed*/
40     int            defaultchar;      /* default char (not glyph index)*/
41     int32_t        bits_size;        /* # words of MWIMAGEBITS bits*/
42 } font_descriptor_t;
43
44 extern font_descriptor_t font_winFreeSystem14x16;
45
46 extern font_descriptor_t font_rom8x16;
47
48 #ifdef __cplusplus
49 } /* extern "C"*/
50 #endif
51
52 #endif /*FONT_TYPES_H*/

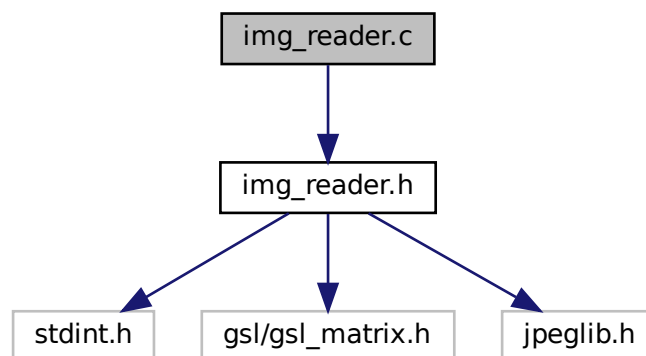
```

3.5 img_reader.c File Reference

Image reading and saving functions using JPEG format.

```
#include "img_reader.h"
```

Include dependency graph for img_reader.c:



Functions

- `gsl_matrix * read_image (const char *filename, int *width, int *height)`
Read an image from a file and convert it to grayscale.
- `void save_image (const gsl_matrix *image, const char *filename)`
Save a grayscale image to a file.

3.5.1 Detailed Description

Image reading and saving functions using JPEG format.

3.5.2 Function Documentation

3.5.2.1 read_image()

```
gsl_matrix * read_image (
    const char * filename,
    int * width,
    int * height )
```

Read an image from a file and convert it to grayscale.

Parameters

| | |
|-----------------|---|
| <i>filename</i> | The name of the image file. |
| <i>width</i> | Pointer to store the width of the image. |
| <i>height</i> | Pointer to store the height of the image. |

Returns

Pointer to the grayscale image matrix.

3.5.2.2 save_image()

```
void save_image (
    const gsl_matrix * image,
    const char * filename )
```

Save a grayscale image to a file.

Parameters

| | |
|-----------------|------------------------------|
| <i>image</i> | The grayscale image matrix. |
| <i>filename</i> | The name of the output file. |

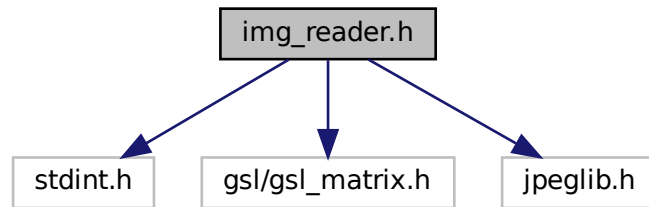
3.6 img_reader.h File Reference

Header file for image reading and saving functions.

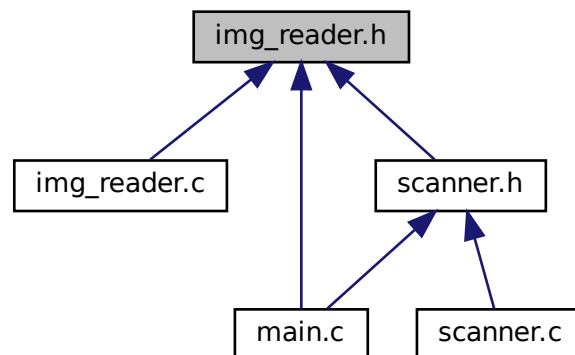
```
#include <stdint.h>
#include <gsl/gsl_matrix.h>
```

```
#include <jpeglib.h>
```

Include dependency graph for img_reader.h:



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



Functions

- `gsl_matrix * read_image` (const char *filename, int *width, int *height)
Read an image from a file and convert it to grayscale.
- void `save_image` (const gsl_matrix *image, const char *filename)
Save a grayscale image to a file.

3.6.1 Detailed Description

Header file for image reading and saving functions.

3.6.2 Function Documentation

3.6.2.1 read_image()

```
gsl_matrix * read_image (
    const char * filename,
    int * width,
    int * height )
```

Read an image from a file and convert it to grayscale.

Parameters

| | |
|-----------------|---|
| <i>filename</i> | The name of the image file. |
| <i>width</i> | Pointer to store the width of the image. |
| <i>height</i> | Pointer to store the height of the image. |

Returns

Pointer to the grayscale image matrix.

3.6.2.2 save_image()

```
void save_image (
    const gsl_matrix * image,
    const char * filename )
```

Save a grayscale image to a file.

Parameters

| | |
|-----------------|------------------------------|
| <i>image</i> | The grayscale image matrix. |
| <i>filename</i> | The name of the output file. |

3.7 img_reader.h

[Go to the documentation of this file.](#)

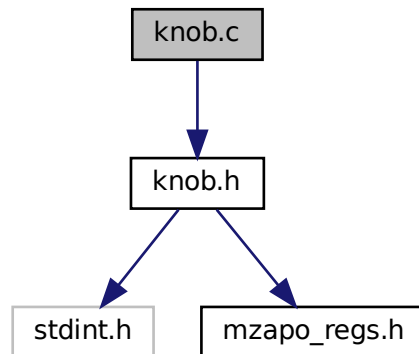
```
1
5 #ifndef IMG_READER_H
6 #define IMG_READER_H
7 #define A4_WIDTH 320
8 #define A4_HEIGHT 480
9
10 #include <stdint.h>
11 #include <gsl/gsl_matrix.h>
12 #include <jpeglib.h>
21 gsl_matrix* read_image(const char *filename, int *width, int *height);
28 void save_image(const gsl_matrix *image, const char *filename);
29
30 #endif /*IMG_READER_H*/
```

3.8 knob.c File Reference

Implementation of knob value reading functions.

```
#include "knob.h"
```

Include dependency graph for knob.c:



Functions

- `int8_t get_knob_value` (unsigned char *spiled_base, const uint8_t knob, const int current)
Get the value of a knob.

3.8.1 Detailed Description

Implementation of knob value reading functions.

3.8.2 Function Documentation

3.8.2.1 get_knob_value()

```
int8_t get_knob_value (
    unsigned char * spiled_base,
    const uint8_t knob,
    const int current )
```

Get the value of a knob.

Parameters

| | |
|--------------------|--------------------------------|
| <i>spiled_base</i> | Base address of the SPILED. |
| <i>knob</i> | The knob identifier. |
| <i>current</i> | The current value of the knob. |

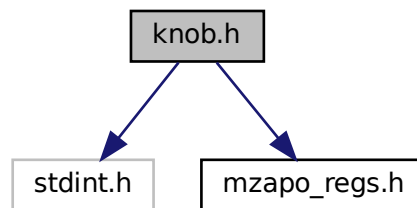
Returns

The difference between the knob value and the current value.

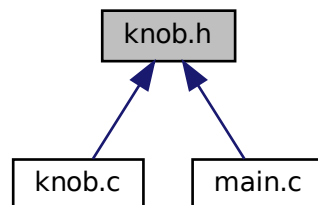
3.9 knob.h File Reference

Header file for knob value reading functions.

```
#include <stdint.h>
#include "mzap_regs.h"
Include dependency graph for knob.h:
```



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

**Functions**

- `int8_t get_knob_value` (unsigned char *spiled_base, const uint8_t knob, const int current)
Get the value of a knob.

3.9.1 Detailed Description

Header file for knob value reading functions.

3.9.2 Function Documentation

3.9.2.1 get_knob_value()

```
int8_t get_knob_value (
    unsigned char * spiled_base,
    const uint8_t knob,
    const int current )
```

Get the value of a knob.

Parameters

| | |
|--------------------|--------------------------------|
| <i>spiled_base</i> | Base address of the SPILED. |
| <i>knob</i> | The knob identifier. |
| <i>current</i> | The current value of the knob. |

Returns

The difference between the knob value and the current value.

3.10 knob.h

[Go to the documentation of this file.](#)

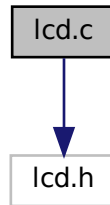
```
1
5 #ifndef KNOB_H
6 #define KNOB_H
7
8 #include <stdint.h>
9 #include "mzap_regs.h"
10
11 #define BLUE_KNOB SPILED_REG_KNOBS_8BIT_o
12 #define GREEN_KNOB SPILED_REG_KNOBS_8BIT_o + 1
13 #define RED_KNOB SPILED_REG_KNOBS_8BIT_o + 2
22 int8_t get_knob_value(unsigned char *spiled_base, const uint8_t knob, const int current);
23
24
25 #endif // KNOB_H
```

3.11 lcd.c File Reference

Implementation of LCD display functions.

```
#include "lcd.h"
```

Include dependency graph for lcd.c:



Functions

- unsigned short * [lcd_init](#) (unsigned char *parlcd_mem_base)
Initialize the LCD display.
- void [lcd_draw_pixel](#) (unsigned short *fb, int x, int y, unsigned short color)
Draw a pixel on the LCD.
- void [lcd_draw_plus](#) (unsigned short *fb, int x, int y, unsigned short color)
Draw a plus sign on the LCD.
- void [lcd_update_display](#) (unsigned short *fb, unsigned char *parlcd_mem_base)
Update the LCD display with the framebuffer content.
- unsigned short [lcd_color](#) (uint8_t red, uint8_t green, uint8_t blue)
Convert RGB values to a 16-bit color.
- unsigned short [lcd_grey](#) (uint8_t intensity)
Convert an 8-bit grayscale intensity to a 16-bit color.
- void [lcd_draw_image](#) (unsigned short *fb, int width, int height, gsl_matrix *image)
Draw an image on the LCD.
- void [lcd_fill_screen](#) (unsigned short *fb, unsigned short color)
Fill the entire screen with a specific color.
- void [lcd_draw_char](#) (unsigned short *fb, int x, int y, font_descriptor_t *fdes, char ch, unsigned short color, unsigned int font_size)
Draw a character on the LCD.
- int [lcd_char_width](#) (char ch, font_descriptor_t *fdes)
Get the width of a character in a specific font.
- void [lcd_draw_scaled_pixel_block](#) (unsigned short *fb, int x, int y, unsigned short color, unsigned int scale)
Draw a block of scaled pixels on the LCD.
- void [lcd_draw_char_bitmap](#) (unsigned short *fb, const font_bits_t *ptr, int x, int y, int w, int h, unsigned short color, unsigned int font_size)
Draw a bitmap of a character on the LCD.
- void [lcd_draw_text](#) (unsigned short *fb, int x, int y, font_descriptor_t *fdes, char *text, unsigned short color, unsigned int font_size)
Draw text on the LCD.

3.11.1 Detailed Description

Implementation of LCD display functions.

3.11.2 Function Documentation

3.11.2.1 lcd_char_width()

```
int lcd_char_width (
    char ch,
    font_descriptor_t * fdes )
```

Get the width of a character in a specific font.

Parameters

| | |
|-------------|---------------------------------|
| <i>ch</i> | The character. |
| <i>fdes</i> | Pointer to the font descriptor. |

Returns

The width of the character.

3.11.2.2 lcd_color()

```
unsigned short lcd_color (
    uint8_t red,
    uint8_t green,
    uint8_t blue )
```

Convert RGB values to a 16-bit color.

Parameters

| | |
|--------------|----------------------|
| <i>red</i> | The red component. |
| <i>green</i> | The green component. |
| <i>blue</i> | The blue component. |

Returns

The 16-bit color value.

3.11.2.3 lcd_draw_char()

```
void lcd_draw_char (
    unsigned short * fb,
```

```

    int x,
    int y,
    font_descriptor_t * fdes,
    char ch,
    unsigned short color,
    unsigned int font_size )

```

Draw a character on the LCD.

Parameters

| | |
|------------------|------------------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>x</i> | The x-coordinate of the character. |
| <i>y</i> | The y-coordinate of the character. |
| <i>fdes</i> | Pointer to the font descriptor. |
| <i>ch</i> | The character to draw. |
| <i>color</i> | The color of the character. |
| <i>font_size</i> | The size of the font. |

3.11.2.4 lcd_draw_char_bitmap()

```

void lcd_draw_char_bitmap (
    unsigned short * fb,
    const font_bits_t * ptr,
    int x,
    int y,
    int w,
    int h,
    unsigned short color,
    unsigned int font_size )

```

Draw a bitmap of a character on the LCD.

Parameters

| | |
|------------------|---------------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>ptr</i> | Pointer to the bitmap data. |
| <i>x</i> | The x-coordinate of the bitmap. |
| <i>y</i> | The y-coordinate of the bitmap. |
| <i>w</i> | The width of the bitmap. |
| <i>h</i> | The height of the bitmap. |
| <i>color</i> | The color of the bitmap. |
| <i>font_size</i> | The size of the font. |

3.11.2.5 lcd_draw_image()

```

void lcd_draw_image (
    unsigned short * fb,

```

```
int width,  
int height,  
gsl_matrix * image )
```

Draw an image on the LCD.

Parameters

| | |
|---------------|-----------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>width</i> | The width of the image. |
| <i>height</i> | The height of the image. |
| <i>image</i> | The grayscale image matrix. |

3.11.2.6 lcd_draw_pixel()

```
void lcd_draw_pixel (  
    unsigned short * fb,  
    int x,  
    int y,  
    unsigned short color )
```

Draw a pixel on the LCD.

Parameters

| | |
|--------------|--------------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>x</i> | The x-coordinate of the pixel. |
| <i>y</i> | The y-coordinate of the pixel. |
| <i>color</i> | The color of the pixel. |

3.11.2.7 lcd_draw_plus()

```
void lcd_draw_plus (  
    unsigned short * fb,  
    int x,  
    int y,  
    unsigned short color )
```

Draw a plus sign on the LCD.

Parameters

| | |
|--------------|---------------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>x</i> | The x-coordinate of the center. |
| <i>y</i> | The y-coordinate of the center. |
| <i>color</i> | The color of the plus sign. |

3.11.2.8 lcd_draw_scaled_pixel_block()

```
void lcd_draw_scaled_pixel_block (
    unsigned short * fb,
    int x,
    int y,
    unsigned short color,
    unsigned int scale )
```

Draw a block of scaled pixels on the LCD.

Parameters

| | |
|--------------|--------------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>x</i> | The x-coordinate of the block. |
| <i>y</i> | The y-coordinate of the block. |
| <i>color</i> | The color of the block. |
| <i>scale</i> | The scaling factor. |

3.11.2.9 lcd_draw_text()

```
void lcd_draw_text (
    unsigned short * fb,
    int x,
    int y,
    font_descriptor_t * fdes,
    char * text,
    unsigned short color,
    unsigned int font_size )
```

Draw text on the LCD.

Parameters

| | |
|------------------|---------------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>x</i> | The x-coordinate of the text. |
| <i>y</i> | The y-coordinate of the text. |
| <i>fdes</i> | Pointer to the font descriptor. |
| <i>text</i> | The text to draw. |
| <i>color</i> | The color of the text. |
| <i>font_size</i> | The size of the font. |

3.11.2.10 lcd_fill_screen()

```
void lcd_fill_screen (
    unsigned short * fb,
    unsigned short color )
```

Fill the entire screen with a specific color.

Parameters

| | |
|--------------|------------------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>color</i> | The color to fill the screen with. |

3.11.2.11 lcd_grey()

```
unsigned short lcd_grey (
    uint8_t intensity )
```

Convert an 8-bit grayscale intensity to a 16-bit color.

Parameters

| | |
|------------------|--------------------------|
| <i>intensity</i> | The grayscale intensity. |
|------------------|--------------------------|

Returns

The 16-bit color value.

3.11.2.12 lcd_init()

```
unsigned short * lcd_init (
    unsigned char * parlcd_mem_base )
```

Initialize the LCD display.

Parameters

| | |
|------------------------|---------------------------------|
| <i>parlcd_mem_base</i> | Base address of the LCD memory. |
|------------------------|---------------------------------|

Returns

Pointer to the framebuffer.

3.11.2.13 lcd_update_display()

```
void lcd_update_display (
    unsigned short * fb,
    unsigned char * parlcd_mem_base )
```

Update the LCD display with the framebuffer content.

Parameters

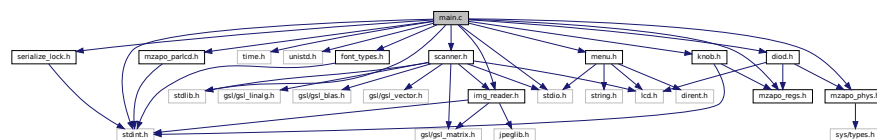
| | |
|------------------------|---------------------------------|
| <i>fb</i> | Pointer to the framebuffer. |
| <i>parlcd_mem_base</i> | Base address of the LCD memory. |

3.12 main.c File Reference

Main file for the MicroZed based MZ_APO board project.

```
#include <stdlib.h>
#include <stdio.h>
#include <stdint.h>
#include <time.h>
#include <unistd.h>
#include "mzapo_parlcd.h"
#include "font_types.h"
#include "mzapo_phys.h"
#include "mzapo_regs.h"
#include "serialize_lock.h"
#include "scanner.h"
#include "img_reader.h"
#include "menu.h"
#include "knob.h"
#include "diod.h"
```

Include dependency graph for main.c:



Functions

- unsigned char * [initialize_memory](#) (uint32_t phys_base, size_t size)
Initialize memory mapping.
- unsigned short * [initialize_display](#) (unsigned char *parlcd_mem_base)
Initialize the display.
- void [load_image](#) (const char *file_path, gsl_matrix **gray_image, int *width, int *height)
Load an image from a file.
- void [transform_image](#) (gsl_matrix *src_mat, gsl_matrix *dst_mat, gsl_matrix *H, int width, int height)

Transform an image using perspective transformation.

- void `capture_points` (uint16_t *xs, uint16_t *ys, unsigned char *spiled_base, unsigned short *fb, gsl_matrix *formatted_image, unsigned char *parlcd_mem_base)

Capture points using the knobs and display.

- void `save_transformed_image` (const gsl_matrix *image, const char *filename)

Save the transformed image to a file.

- void `app_loop` (void)

Application loop.

- int `main` (void)

Main function.

3.12.1 Detailed Description

Main file for the MicroZed based MZ_APO board project.

3.12.2 Function Documentation

3.12.2.1 `capture_points()`

```
void capture_points (
    uint16_t * xs,
    uint16_t * ys,
    unsigned char * spiled_base,
    unsigned short * fb,
    gsl_matrix * formatted_image,
    unsigned char * parlcd_mem_base )
```

Capture points using the knobs and display.

Parameters

| | |
|------------------------------|---|
| <code>xs</code> | Array to store the x-coordinates of the points. |
| <code>ys</code> | Array to store the y-coordinates of the points. |
| <code>spiled_base</code> | Base address of the SPILED. |
| <code>fb</code> | Pointer to the framebuffer. |
| <code>formatted_image</code> | The formatted image matrix. |
| <code>parlcd_mem_base</code> | Base address of the LCD memory. |

3.12.2.2 `initialize_display()`

```
unsigned short * initialize_display (
    unsigned char * parlcd_mem_base )
```

Initialize the display.

Parameters

| | |
|------------------------|---------------------------------|
| <i>parlcd_mem_base</i> | Base address of the LCD memory. |
|------------------------|---------------------------------|

Returns

Pointer to the framebuffer.

3.12.2.3 initialize_memory()

```
unsigned char * initialize_memory (
    uint32_t phys_base,
    size_t size )
```

Initialize memory mapping.

Parameters

| | |
|------------------|--------------------------------|
| <i>phys_base</i> | The physical base address. |
| <i>size</i> | The size of the memory region. |

Returns

Pointer to the mapped memory.

3.12.2.4 load_image()

```
void load_image (
    const char * file_path,
    gsl_matrix ** gray_image,
    int * width,
    int * height )
```

Load an image from a file.

Parameters

| | |
|-------------------|--|
| <i>file_path</i> | The path to the image file. |
| <i>gray_image</i> | Pointer to the grayscale image matrix. |
| <i>width</i> | Pointer to the width of the image. |
| <i>height</i> | Pointer to the height of the image. |

3.12.2.5 main()

```
int main (
    void )
```

Main function.

Returns

Exit status.

3.12.2.6 save_transformed_image()

```
void save_transformed_image (
    const gsl_matrix * image,
    const char * filename )
```

Save the transformed image to a file.

Parameters

| | |
|-----------------|-------------------------------|
| <i>image</i> | The transformed image matrix. |
| <i>filename</i> | The name of the output file. |

3.12.2.7 transform_image()

```
void transform_image (
    gsl_matrix * src_mat,
    gsl_matrix * dst_mat,
    gsl_matrix * H,
    int width,
    int height )
```

Transform an image using perspective transformation.

Parameters

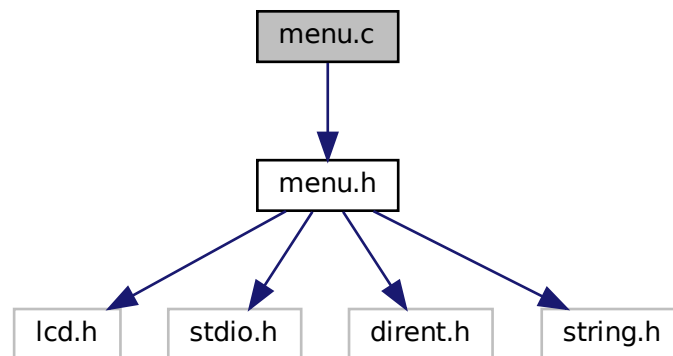
| | |
|----------------|-------------------------------|
| <i>src_mat</i> | Source matrix of points. |
| <i>dst_mat</i> | Destination matrix of points. |
| <i>H</i> | Homography matrix. |
| <i>width</i> | The width of the image. |
| <i>height</i> | The height of the image. |

3.13 menu.c File Reference

Implementation of menu functions.

```
#include "menu.h"
```

Include dependency graph for menu.c:



Functions

- void **print_dir** (void)
Print all images in the directory.
- void **get_file_name** (char *file_name, int file_number)
Get the name of a file by its number in the list.

3.13.1 Detailed Description

Implementation of menu functions.

3.13.2 Function Documentation

3.13.2.1 get_file_name()

```
void get_file_name (  
    char * file_name,  
    int file_number )
```

Get the name of a file by its number in the list.

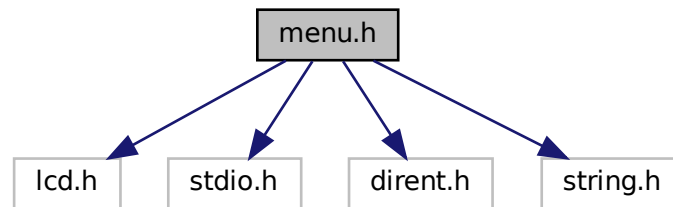
Parameters

| | |
|--------------------|-------------------------------------|
| <i>file_name</i> | Buffer to store the file name. |
| <i>file_number</i> | The number of the file in the list. |

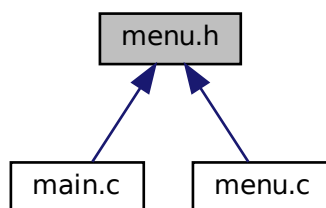
3.14 menu.h File Reference

Header file for menu functions.

```
#include "lcd.h"  
#include <stdio.h>  
#include <dirent.h>  
#include <string.h>  
Include dependency graph for menu.h:
```



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



Functions

- void **show_menu** (void)
Display the menu.
- void **print_dir** (void)
Print all images in the directory.
- void **get_file_name** (char *file_name, int file_number)
Get the name of a file by its number in the list.

3.14.1 Detailed Description

Header file for menu functions.

3.14.2 Function Documentation

3.14.2.1 `get_file_name()`

```
void get_file_name (
    char * file_name,
    int file_number )
```

Get the name of a file by its number in the list.

Parameters

| | |
|--------------------|-------------------------------------|
| <i>file_name</i> | Buffer to store the file name. |
| <i>file_number</i> | The number of the file in the list. |

3.15 menu.h

[Go to the documentation of this file.](#)

```
1
6 #ifndef MENU_H
7 #define MENU_H
8
9 #include "lcd.h"
10 #include <stdio.h>
11 #include <dirent.h>
12 #include <string.h>
13 #define APP_DIR "/tmp/filkiana/"
14
18 void show_menu(void);
19
23 void print_dir(void);
24
31 void get_file_name(char *file_name, int file_number);
32
33 #endif // MENU_H
```

3.16 mzap0_parlcd.h

```
1 /*****
2 Simple program to check LCD functionality on MicroZed
3 based MZ_APO board designed by Petr Porazil at PiKRON
4
5 mzap0_parlcd.h      - parallel connected LCD low level access
6
7 (C) Copyright 2017 by Pavel Pisa
8 e-mail:      pisa@cmp.felk.cvut.cz
9 homepage:    http://cmp.felk.cvut.cz/~pisa
10 company:     http://www.pikron.com/
11 license:     any combination of GPL, LGPL, MPL or BSD licenses
12
13 *****/
14
```

```

15 #ifndef MZAPO_PARLCD_H
16 #define MZAPO_PARLCD_H
17
18 #include <stdint.h>
19
20 #ifdef __cplusplus
21 extern "C" {
22 #endif
23
24 void parlcd_write_cr(unsigned char *parlcd_mem_base, uint16_t data);
25
26 void parlcd_write_cmd(unsigned char *parlcd_mem_base, uint16_t cmd);
27
28 void parlcd_write_data(unsigned char *parlcd_mem_base, uint16_t data);
29
30 void parlcd_write_data2x(unsigned char *parlcd_mem_base, uint32_t data);
31
32 void parlcd_delay(int msec);
33
34 void parlcd_hx8357_init(unsigned char *parlcd_mem_base);
35
36
37 #ifdef __cplusplus
38 } /* extern "C" */
39 #endif
40
41 #endif /*MZAPO_PARLCD_H*/

```

3.17 mzap0_phys.h

```

1 /*****
2 Simple program to check LCD functionality on MicroZed
3 based MZ_APO board designed by Petr Porazil at PiKRON
4
5 mzap0_phys.h      - mapping of the physical address to process
6
7 (C) Copyright 2017 by Pavel Pisa
8 e-mail:          pisa@cmp.felk.cvut.cz
9 homepage:       http://cmp.felk.cvut.cz/~pisa
10 company:        http://www.pikron.com/
11 license:        any combination of GPL, LGPL, MPL or BSD licenses
12
13 *****/
14
15 #ifndef MZAPO_PHYS_H
16 #define MZAPO_PHYS_H
17
18 #include <sys/types.h>
19
20 #ifdef __cplusplus
21 extern "C" {
22 #endif
23
24 void *map_phys_address(off_t region_base, size_t region_size, int opt_cached);
25
26 #ifdef __cplusplus
27 } /* extern "C" */
28 #endif
29
30 #endif /*MZAPO_PHYS_H*/

```

3.18 mzap0_regs.h

```

1 /*****
2 Simple program to check LCD functionality on MicroZed
3 based MZ_APO board designed by Petr Porazil at PiKRON
4
5 mzap0_regs.h      - definition of the MZ_APO design registers
6
7 (C) Copyright 2017 by Pavel Pisa
8 e-mail:          pisa@cmp.felk.cvut.cz
9 homepage:       http://cmp.felk.cvut.cz/~pisa
10 company:        http://www.pikron.com/
11 license:        any combination of GPL, LGPL, MPL or BSD licenses
12
13 *****/
14
15 #ifndef MZAPO_REGS_H
16 #define MZAPO_REGS_H

```

```

17
18 /*
19 Complete description of the educational MZ_APO design registers
20 can be found at
21
22 https://cw.fel.cvut.cz/wiki/courses/b35apo/documentation/mz_apo/start
23
24 The peripherals VHDL sources can be found in the repository
25
26 http://rtime.felk.cvut.cz/gitweb/fpga/zynq/canbench-sw.git/tree/refs/heads/microzed_apo:/system/ip
27
28 */
29
30 /* SPI connected knobs and LEDs registers and keyboard */
31
32 #define SPILED_REG_BASE_PHYS 0x43c40000
33 #define SPILED_REG_SIZE 0x00004000
34
35 #define SPILED_REG_LED_LINE_o 0x004
36 #define SPILED_REG_LED_RGB1_o 0x010
37 #define SPILED_REG_LED_RGB2_o 0x014
38 #define SPILED_REG_LED_KBDWR_DIRECT_o 0x018
39
40 #define SPILED_REG_KBDRD_KNOBS_DIRECT_o 0x020
41 #define SPILED_REG_KNOBS_8BIT_o 0x024
42
43 /* Parallel LCD registers */
44
45 #define PARLCD_REG_BASE_PHYS 0x43c00000
46 #define PARLCD_REG_SIZE 0x00004000
47
48 #define PARLCD_REG_CR_o 0x0000
49 #define PARLCD_REG_CR_RESET_m 0x00000002
50 #define PARLCD_REG_CMD_o 0x0008
51 #define PARLCD_REG_DATA_o 0x000C
52
53 /* RC model servos and optional PS2 peripheral */
54
55 #define SERVOPS2_REG_BASE_PHYS 0x43c50000
56 #define SERVOPS2_REG_SIZE 0x4000
57
58 #define SERVOPS2_REG_CR_o 0x0000
59 #define SERVOPS2_REG_PWMPER_o 0x000C
60 #define SERVOPS2_REG_PWM1_o 0x0010
61 #define SERVOPS2_REG_PWM2_o 0x0014
62 #define SERVOPS2_REG_PWM3_o 0x0018
63 #define SERVOPS2_REG_PWM4_o 0x001C
64
65 /* Simple audio PWM output */
66
67 #define AUDIOPWM_REG_BASE_PHYS 0x43c60000
68 #define AUDIOPWM_REG_SIZE 0x4000
69
70 #define AUDIOPWM_REG_CR_o 0x0000
71 #define AUDIOPWM_REG_PWMPER_o 0x0008
72 #define AUDIOPWM_REG_PWM_o 0x000C
73
74 /* Optional DC Motor Simple Driver Peripherals for PSR Subject */
75
76 #define DCSPDRV_REG_BASE_PHYS_0 0x43c20000
77 #define DCSPDRV_REG_BASE_PHYS_1 0x43c30000
78 #define DCSPDRV_REG_SIZE 0x4000
79
80 #define DCSPDRV_REG_CR_o 0x0000
81 #define DCSPDRV_REG_CR_PWM_A_DIRECT_m 0x00000010
82 #define DCSPDRV_REG_CR_PWM_B_DIRECT_m 0x00000020
83 #define DCSPDRV_REG_CR_PWM_ENABLE_m 0x00000040
84 #define DCSPDRV_REG_CR_IRC_RESET_m 0x00000100
85
86 #define DCSPDRV_REG_SR_o 0x0004
87 #define DCSPDRV_REG_SR_IRC_A_MON_m 0x00000100
88 #define DCSPDRV_REG_SR_IRC_B_MON_m 0x00000200
89 #define DCSPDRV_REG_SR_IRC_IRQ_MON_m 0x00000400
90
91 #define DCSPDRV_REG_PERIOD_o 0x0008
92 #define DCSPDRV_REG_PERIOD_MASK_m 0x3fffffff
93
94 #define DCSPDRV_REG_DUTY_o 0x000C
95 #define DCSPDRV_REG_DUTY_MASK_m 0x3fffffff
96 #define DCSPDRV_REG_DUTY_DIR_A_m 0x40000000
97 #define DCSPDRV_REG_DUTY_DIR_B_m 0x80000000
98
99 #define DCSPDRV_REG_IRC_o 0x0010
100
101 #endif /* MZAPO_REGS_H */

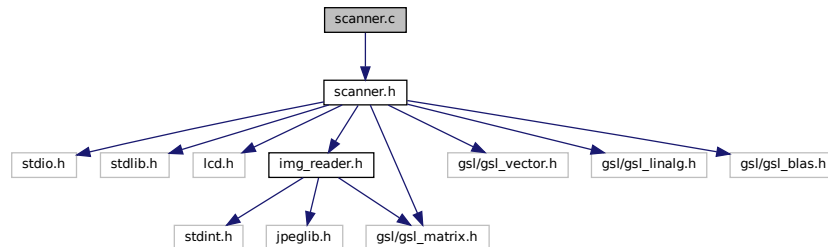
```

3.19 scanner.c File Reference

Implementation of matrix and transformation functions.

```
#include "scanner.h"
```

Include dependency graph for scanner.c:



Functions

- void [print_matrix](#) (const gsl_matrix *m, const char *name)
Print a matrix to the console.
- void [compute_perspective_transform](#) (const gsl_matrix *src, const gsl_matrix *dst, gsl_matrix *H)
Compute the perspective transform matrix.
- void [apply_perspective_transform](#) (const gsl_matrix *gray_image, gsl_matrix *H, gsl_matrix *image_wrapped, int width, int height)
Apply a perspective transform to an image.

3.19.1 Detailed Description

Implementation of matrix and transformation functions.

3.19.2 Function Documentation

3.19.2.1 [apply_perspective_transform\(\)](#)

```
void apply_perspective_transform (
    const gsl_matrix * gray_image,
    gsl_matrix * H,
    gsl_matrix * image_wrapped,
    int width,
    int height )
```

Apply a perspective transform to an image.

Parameters

| | |
|----------------------|------------------------------------|
| <i>gray_image</i> | The source grayscale image matrix. |
| <i>H</i> | The homography matrix. |
| <i>image_wrapped</i> | The destination image matrix. |
| <i>width</i> | The width of the source image. |
| <i>height</i> | The height of the source image. |

3.19.2.2 compute_perspective_transform()

```
void compute_perspective_transform (
    const gsl_matrix * src,
    const gsl_matrix * dst,
    gsl_matrix * H )
```

Compute the perspective transform matrix.

Parameters

| | |
|------------|--------------------------------|
| <i>src</i> | The source points matrix. |
| <i>dst</i> | The destination points matrix. |
| <i>H</i> | The homography matrix. |

3.19.2.3 print_matrix()

```
void print_matrix (
    const gsl_matrix * m,
    const char * name )
```

Print a matrix to the console.

Parameters

| | |
|-------------|-------------------------|
| <i>m</i> | The matrix to print. |
| <i>name</i> | The name of the matrix. |

3.20 scanner.h File Reference

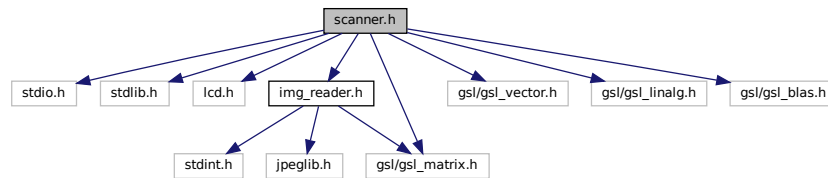
Header file for matrix and transformation functions.

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

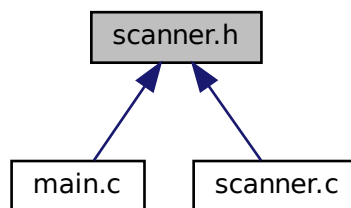


```
#include "lcd.h"
#include "img_reader.h"
#include <gsl/gsl_matrix.h>
#include <gsl/gsl_vector.h>
#include <gsl/gsl_linalg.h>
#include <gsl/gsl_blas.h>
```

Include dependency graph for scanner.h:



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



Functions

- void [print_matrix](#) (const gsl_matrix *m, const char *name)
Print a matrix to the console.
- void [compute_perspective_transform](#) (const gsl_matrix *src, const gsl_matrix *dst, gsl_matrix *H)
Compute the perspective transform matrix.
- void [apply_perspective_transform](#) (const gsl_matrix *gray_image, gsl_matrix *H, gsl_matrix *image_↵ wrapped, int width, int height)
Apply a perspective transform to an image.

3.20.1 Detailed Description

Header file for matrix and transformation functions.

3.20.2 Function Documentation

3.20.2.1 `apply_perspective_transform()`

```
void apply_perspective_transform (
    const gsl_matrix * gray_image,
    gsl_matrix * H,
    gsl_matrix * image_wrapped,
    int width,
    int height )
```

Apply a perspective transform to an image.

Parameters

| | |
|----------------------|------------------------------------|
| <i>gray_image</i> | The source grayscale image matrix. |
| <i>H</i> | The homography matrix. |
| <i>image_wrapped</i> | The destination image matrix. |
| <i>width</i> | The width of the source image. |
| <i>height</i> | The height of the source image. |

3.20.2.2 `compute_perspective_transform()`

```
void compute_perspective_transform (
    const gsl_matrix * src,
    const gsl_matrix * dst,
    gsl_matrix * H )
```

Compute the perspective transform matrix.

Parameters

| | |
|------------|--------------------------------|
| <i>src</i> | The source points matrix. |
| <i>dst</i> | The destination points matrix. |
| <i>H</i> | The homography matrix. |

3.20.2.3 `print_matrix()`

```
void print_matrix (
    const gsl_matrix * m,
    const char * name )
```

Print a matrix to the console.

Parameters

| | |
|-------------|-------------------------|
| <i>m</i> | The matrix to print. |
| <i>name</i> | The name of the matrix. |

3.21 scanner.h

[Go to the documentation of this file.](#)

```
1
6 #ifndef SCANNER_H
7 #define SCANNER_H
8
9 #include <stdio.h>
10 #include <stdlib.h>
11
12 #include "lcd.h"
13 #include "img_reader.h"
14 #include <gsl/gsl_matrix.h>
15 #include <gsl/gsl_vector.h>
16 #include <gsl/gsl_linalg.h>
17 #include <gsl/gsl_blas.h>
18
25 void print_matrix(const gsl_matrix *m, const char *name);
26
34 void compute_perspective_transform(const gsl_matrix *src, const gsl_matrix *dst, gsl_matrix *H);
35
45 void apply_perspective_transform(const gsl_matrix *gray_image, gsl_matrix *H, gsl_matrix *image_wrapped,
    int width, int height);
46
47 #endif /* SCANNER_H */
```

3.22 serialize_lock.h

```
1 #ifndef SERIALIZE_LOCK_H
2 #define SERIALIZE_LOCK_H
3
4 #include <stdint.h>
5
6 #ifdef __cplusplus
7 extern "C" {
8 #endif
9
10 int serialize_lock(int no_wait);
11
12 void serialize_unlock(void);
13
14 #ifdef __cplusplus
15 } /* extern "C" */
16 #endif
17
18 #endif /* SERIALIZE_LOCK_H */
```


Index

- apply_perspective_transform
 - scanner.c, [33](#)
 - scanner.h, [35](#)
- capture_points
 - main.c, [25](#)
- compute_perspective_transform
 - scanner.c, [34](#)
 - scanner.h, [36](#)
- diod.c, [7](#)
 - diod_set_color, [8](#)
- diod.h, [8](#)
 - diod_set_color, [9](#)
- diod_set_color
 - diod.c, [8](#)
 - diod.h, [9](#)
- get_file_name
 - menu.c, [28](#)
 - menu.h, [30](#)
- get_knob_value
 - knob.c, [15](#)
 - knob.h, [17](#)
- img_reader.c, [11](#)
 - read_image, [12](#)
 - save_image, [12](#)
- img_reader.h, [12](#)
 - read_image, [13](#)
 - save_image, [14](#)
- initialize_display
 - main.c, [25](#)
- initialize_memory
 - main.c, [26](#)
- knob.c, [15](#)
 - get_knob_value, [15](#)
- knob.h, [16](#)
 - get_knob_value, [17](#)
- lcd.c, [17](#)
 - lcd_char_width, [19](#)
 - lcd_color, [19](#)
 - lcd_draw_char, [19](#)
 - lcd_draw_char_bitmap, [20](#)
 - lcd_draw_image, [20](#)
 - lcd_draw_pixel, [21](#)
 - lcd_draw_plus, [21](#)
 - lcd_draw_scaled_pixel_block, [22](#)
 - lcd_draw_text, [22](#)
 - lcd_fill_screen, [22](#)
 - lcd_grey, [23](#)
 - lcd_init, [23](#)
 - lcd_update_display, [23](#)
- lcd_char_width
 - lcd.c, [19](#)
- lcd_color
 - lcd.c, [19](#)
- lcd_draw_char
 - lcd.c, [19](#)
- lcd_draw_char_bitmap
 - lcd.c, [20](#)
- lcd_draw_image
 - lcd.c, [20](#)
- lcd_draw_pixel
 - lcd.c, [21](#)
- lcd_draw_plus
 - lcd.c, [21](#)
- lcd_draw_scaled_pixel_block
 - lcd.c, [22](#)
- lcd_draw_text
 - lcd.c, [22](#)
- lcd_fill_screen
 - lcd.c, [22](#)
- lcd_grey
 - lcd.c, [23](#)
- lcd_init
 - lcd.c, [23](#)
- lcd_update_display
 - lcd.c, [23](#)
- load_image
 - main.c, [26](#)
- main
 - main.c, [26](#)
- main.c, [24](#)
 - capture_points, [25](#)
 - initialize_display, [25](#)
 - initialize_memory, [26](#)
 - load_image, [26](#)
 - main, [26](#)
 - save_transformed_image, [27](#)
 - transform_image, [27](#)
- menu.c, [28](#)
 - get_file_name, [28](#)
- menu.h, [29](#)
 - get_file_name, [30](#)
- print_matrix
 - scanner.c, [34](#)

- scanner.h, [36](#)
- read_image
 - img_reader.c, [12](#)
 - img_reader.h, [13](#)
- save_image
 - img_reader.c, [12](#)
 - img_reader.h, [14](#)
- save_transformed_image
 - main.c, [27](#)
- scanner.c, [33](#)
 - apply_perspective_transform, [33](#)
 - compute_perspective_transform, [34](#)
 - print_matrix, [34](#)
- scanner.h, [34](#)
 - apply_perspective_transform, [35](#)
 - compute_perspective_transform, [36](#)
 - print_matrix, [36](#)
- transform_image
 - main.c, [27](#)