My Project

Generated by Doxygen 1.9.4

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 intencity 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

0. **Download and build the dependencies:**

- https://www.gnu.org/software/gsl/
- https://www.ijg.org/

1. For LGSL: gsl.tar.gz ```bash tar -xvzf gsl.tar.gz cd gsl ./configure "CC=arm-linux-gnueabihf-gcc" make ```
2. For LJPEG: jpegsrc.v9d.tar.gz ```bash tar -xvzf jpegsrc.v9d.tar.gz cd jpeg-9d ./configure "CC=arm-linux-gnueabihf-gcc" –prefix={DIR_WHERE_YOU_HAVE_THIS_LIBRARY} make
```
1.  **Clone the Repository:**
   "'bash
   git clone https://github.com/filkiana/APO_SEM
   cd APO_SEM
```

Configure Makefile with the paths to the dependencies and board ip:
```
using your favorite text editor.
```

1. **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.

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

Class Documentation

4.1 font_descriptor_t Struct Reference

Public Attributes

- char * **name**
- int **maxwidth**
- unsigned int **height**
- int **ascent**
- int **firstchar**
- int **size**
- const font_bits_t * **bits**
- const uint32_t * **offset**
- const unsigned char * **width**
- int **defaultchar**
- int32_t **bits_size**

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

- font_types.h

Chapter 5

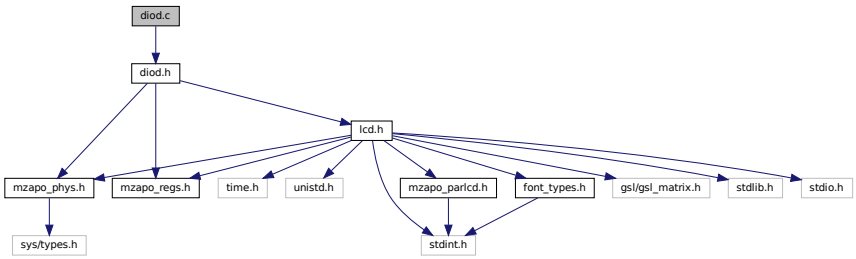
File Documentation

5.1 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.

5.1.1 Detailed Description

Implementation of Diod LED control functions.

5.1.2 Function Documentation

5.1.2.1 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.

Set the color of the LED.

Parameters

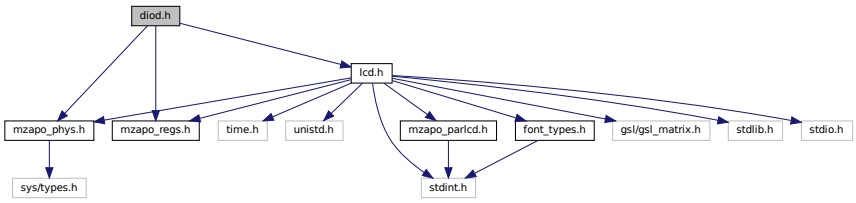
spiled_base	Base address of the SPILED.
r	Red component of the color.
g	Green component of the color.
b	Blue component of the color.

5.2 diod.h File Reference

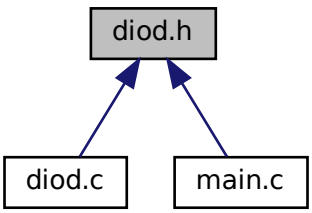
Header file for LED control functions.

```
#include "mzapo_phys.h"
#include "mzapo_regs.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 LED.

5.2.1 Detailed Description

Header file for LED control functions.

5.2.2 Function Documentation

5.2.2.1 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 LED.

Parameters

spiled_base	Base address of the SPILED.
r	Red component of the color.
g	Green component of the color.
b	Blue component of the color.

Set the color of the LED.

Parameters

spiled_base	Base address of the SPILED.
r	Red component of the color.
g	Green component of the color.
b	Blue component of the color.

5.3 diod.h

Go to the documentation of this file.

```
1
5 #ifndef DIOD_H
6 #define DIOD_H
7 #include "mzapo_phys.h"
8 #include "mzapo_regs.h"
```

```
9 #include "lcd.h"
10
19 void diod_set_color(unsigned char *spiled_base, unsigned char r, unsigned char g, unsigned char b);
20
21 #endif // DIOD_H
```

5.4 font_types.h

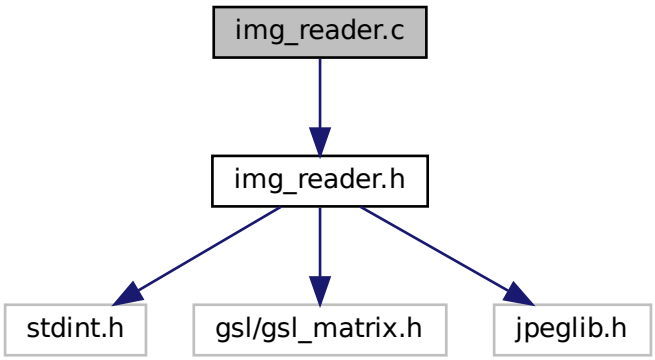
```
1 /*********************************************************************
2
3 font_types.h      - simple bitmap fonts type definition
4
5 Simplified font type descriptor based on
6 Microwindows/Nano-X library by Greg Haerr
7
8 https://github.com/ghaerr/microwindows
9
10 Copyright (c) 1999, 2000, 2001, 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*/
```

5.5 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.

5.5.1 Detailed Description

Image reading and saving functions using JPEG format.

5.5.2 Function Documentation

5.5.2.1 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

filename	The name of the image file.
width	Pointer to store the width of the image.
height	Pointer to store the height of the image.

Returns

Pointer to the grayscale image matrix.

5.5.2.2 save_image()

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

Save a grayscale image to a file.

Parameters

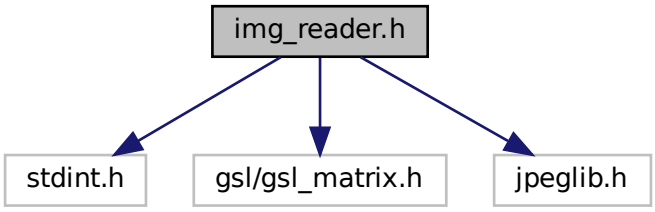
image	The grayscale image matrix.
filename	The name of the output file.

5.6 img_reader.h File Reference

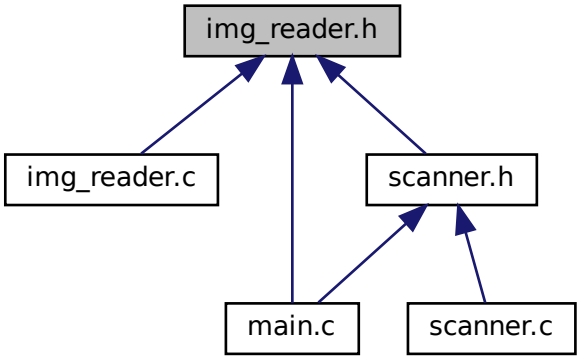
Header file for image reading and saving functions.

```
#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:



Macros

- #define **A4_WIDTH** 320
- #define **A4_HEIGHT** 480

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.

5.6.1 Detailed Description

Header file for image reading and saving functions.

5.6.2 Function Documentation

5.6.2.1 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

filename	The name of the image file.
width	Pointer to store the width of the image.
height	Pointer to store the height of the image.

Returns

Pointer to the grayscale image matrix.

5.6.2.2 save_image()

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

Save a grayscale image to a file.

Parameters

image	The grayscale image matrix.
filename	The name of the output file.

5.7 img_reader.h

Go to the documentation of this file.

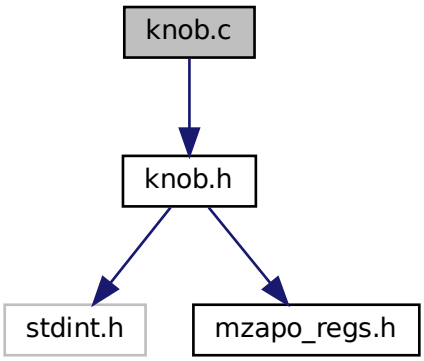
```
1
5 #ifndef IMG_READER_H
6 #define IMG_READER_H
7 #define A4_WIDTH 320
8 #define A4_HEIGHT 480
9
10 #include <stdint.h>
11 #include <gsl/gsl_matrix.h>
12 #include <jpeglib.h>
21 gsl_matrix* read_image(const char *filename, int *width, int *height);
28 void save_image(const gsl_matrix *image, const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

5.8 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.

5.8.1 Detailed Description

Implementation of knob value reading functions.

5.8.2 Function Documentation

5.8.2.1 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

spiled_base	Base address of the SPILED.
knob	The knob identifier.
current	The current value of the knob.

Returns

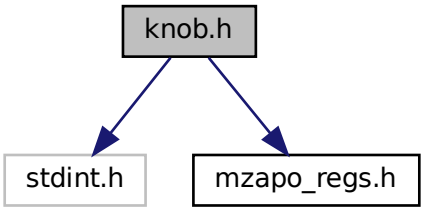
The difference between the knob value and the current value.

5.9 knob.h File Reference

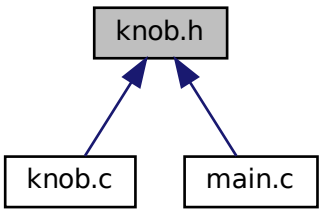
Header file for knob value reading functions.

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

Include dependency graph for knob.h:



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



Macros

- #define **BLUE_KNOB** SPILED_REG_KNOBS_8BIT_o
- #define **GREEN_KNOB** SPILED_REG_KNOBS_8BIT_o + 1
- #define **RED_KNOB** SPILED_REG_KNOBS_8BIT_o + 2

Functions

- int8_t get_knob_value (unsigned char ∗spiled_base, const uint8_t knob, const int current)

  Get the value of a knob.

5.9.1 Detailed Description

Header file for knob value reading functions.

5.9.2 Function Documentation

5.9.2.1 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

spiled_base	Base address of the SPILED.
knob	The knob identifier.
current	The current value of the knob.

Returns

The difference between the knob value and the current value.

5.10 knob.h

Go to the documentation of this file.

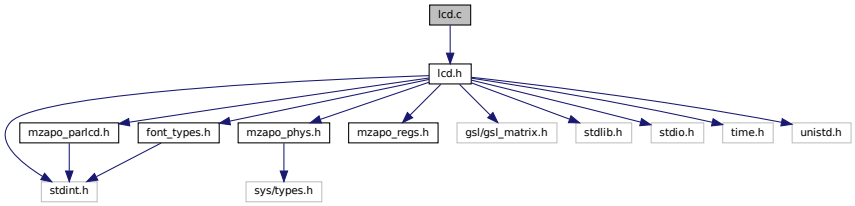
```
1
5 #ifndef KNOB_H
6 #define KNOB_H
7
8 #include <stdint.h>
9 #include "mzapo_regs.h"
10
11 #define BLUE_KNOB SPILED_REG_KNOBS_8BIT_o
12 #define GREEN_KNOB SPILED_REG_KNOBS_8BIT_o + 1
13 #define RED_KNOB SPILED_REG_KNOBS_8BIT_o + 2
22 int8_t get_knob_value(unsigned char *spiled_base,const uint8_t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

5.11 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 current_x, int current_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.

5.11.1 Detailed Description

Implementation of LCD display functions.

5.11.2 Function Documentation

5.11.2.1 lcd_char_width()

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

Get the width of a character in a specific font.

Parameters

ch	The character.
fdes	Pointer to the font descriptor.

Returns

The width of the character.

5.11.2.2 lcd_color()

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

Convert RGB values to a 16-bit color.

Parameters

red	The red component.
green	The green component.
blue	The blue component.

Returns

The 16-bit color value.

5.11.2.3 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

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

5.11.2.4 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

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

5.11.2.5 lcd_draw_image()

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

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

Draw an image on the LCD.

Parameters

fb	Pointer to the framebuffer.
width	The width of the image.
height	The height of the image.
image	The grayscale image matrix.

5.11.2.6 lcd_draw_pixel()

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

Draw a pixel on the LCD.

Parameters

fb	Pointer to the framebuffer.
x	The x-coordinate of the pixel.
y	The y-coordinate of the pixel.
color	The color of the pixel.

5.11.2.7 lcd_draw_plus()

```
void lcd_draw_plus (
            unsigned short * fb,
            int current_x,
            int current_y,
            unsigned short color )
```

Draw a plus sign on the LCD.

Parameters

fb	Pointer to the framebuffer.
current_x	The x-coordinate of the center.
current_y	The y-coordinate of the center.
color	The color of the plus sign.

5.11.2.8 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

fb	Pointer to the framebuffer.
x	The x-coordinate of the block.
y	The y-coordinate of the block.
color	The color of the block.
scale	The scaling factor.

5.11.2.9 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

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

5.11.2.10 lcd_fill_screen()

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

Fill the entire screen with a specific color.

Parameters

fb	Pointer to the framebuffer.
color	The color to fill the screen with.

5.11.2.11 lcd_grey()

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

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

Parameters

intensity	The grayscale intensity.

Returns

The 16-bit color value.

5.11.2.12 lcd_init()

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

Initialize the LCD display.

Parameters

parlcd_mem_base	Base address of the LCD memory.

Returns

Pointer to the framebuffer.

5.11.2.13 lcd_update_display()

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

Update the LCD display with the framebuffer content.

Parameters

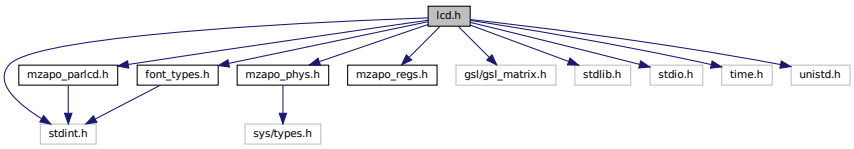
fb	Pointer to the framebuffer.
parlcd_mem_base	Base address of the LCD memory.

5.12 lcd.h File Reference

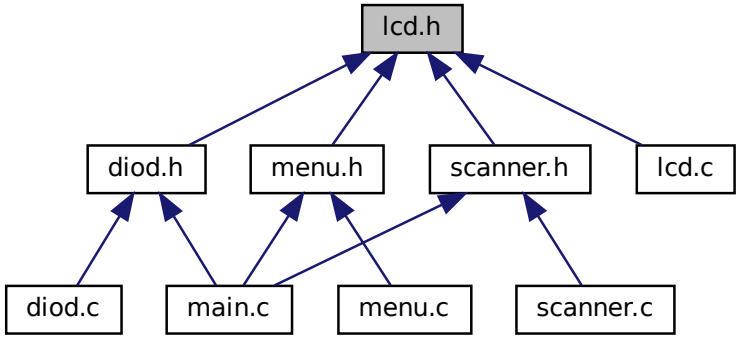
Header file for LCD display functions.

```
#include <stdint.h>
#include "mzapo_parlcd.h"
#include "mzapo_phys.h"
#include "font_types.h"
#include "mzapo_regs.h"
#include <gsl/gsl_matrix.h>
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <unistd.h>
```

Include dependency graph for lcd.h:



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



Macros

- #define **HEIGHT** 320
- #define **WIDTH** 480

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_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.
- void lcd_update_display (unsigned short *fb, unsigned char *parlcd_mem_base)

  Update the LCD display with the framebuffer content.
- unsigned short lcd_grey (uint8_t intensity)

  Convert an 8-bit grayscale intensity to a 16-bit color.
- unsigned short lcd_color (uint8_t red, uint8_t green, uint8_t blue)

  Convert RGB values 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_draw_plus (unsigned short *fb, int x, int y, unsigned short color)

  Draw a plus sign on the LCD.

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

  Fill the entire screen with a specific color.

5.12.1 Detailed Description

Header file for LCD display functions.

5.12.2 Function Documentation

5.12.2.1 lcd_char_width()

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

Get the width of a character in a specific font.

Parameters

ch	The character.
fdes	Pointer to the font descriptor.

Returns

The width of the character.

5.12.2.2 lcd_color()

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

Convert RGB values to a 16-bit color.

Parameters

red	The red component.
green	The green component.
blue	The blue component.

Returns

The 16-bit color value.

5.12.2.3 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

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

5.12.2.4 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

fb	Pointer to the framebuffer.
ptr	Pointer to the bitmap data.
x	The x-coordinate of the bitmap.
y	The y-coordinate of the bitmap.
w	The width of the bitmap.

Parameters

h	The height of the bitmap.
color	The color of the bitmap.
font_size	The size of the font.

5.12.2.5 lcd_draw_image()

```
void lcd_draw_image (
            unsigned short * fb,
            int width,
            int height,
            gsl_matrix * image )
```

Draw an image on the LCD.

Parameters

fb	Pointer to the framebuffer.
width	The width of the image.
height	The height of the image.
image	The grayscale image matrix.

5.12.2.6 lcd_draw_pixel()

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

Draw a pixel on the LCD.

Parameters

fb	Pointer to the framebuffer.
x	The x-coordinate of the pixel.
y	The y-coordinate of the pixel.
color	The color of the pixel.

5.12.2.7 lcd_draw_plus()

```
void lcd_draw_plus (
```

```
            unsigned short * fb,
            int current_x,
            int current_y,
            unsigned short color )
```

Draw a plus sign on the LCD.

Parameters

fb	Pointer to the framebuffer.
x	The x-coordinate of the center.
y	The y-coordinate of the center.
color	The color of the plus sign.
fb	Pointer to the framebuffer.
current_x	The x-coordinate of the center.
current_y	The y-coordinate of the center.
color	The color of the plus sign.

5.12.2.8 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

fb	Pointer to the framebuffer.
x	The x-coordinate of the block.
y	The y-coordinate of the block.
color	The color of the block.
scale	The scaling factor.

5.12.2.9 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

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

5.12.2.10 lcd_fill_screen()

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

Fill the entire screen with a specific color.

Parameters

fb	Pointer to the framebuffer.
color	The color to fill the screen with.

5.12.2.11 lcd_grey()

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

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

Parameters

intensity	The grayscale intensity.

Returns

The 16-bit color value.

5.12.2.12 lcd_init()

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

Initialize the LCD display.

Parameters

parlcd_mem_base	Base address of the LCD memory.

Returns

Pointer to the framebuffer.

5.12.2.13 lcd_update_display()

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

Update the LCD display with the framebuffer content.

Parameters

fb	Pointer to the framebuffer.
parlcd_mem_base	Base address of the LCD memory.

5.13 lcd.h

Go to the documentation of this file.

```
1
5 #ifndef LCD_H
6 #define LCD_H
7 #include <stdint.h>
8 #include "mzapo_parlcd.h"
9 #include "mzapo_phys.h"
10 #include "font_types.h"
11 #include "mzapo_regs.h"
12
13 #include <gsl/gsl_matrix.h>
14 #include <stdlib.h>
15 #include <stdint.h>
16 #include <stdio.h>
17 #include <time.h>
18 #include <unistd.h>
19
20 #define HEIGHT 320
21 #define WIDTH 480
22
23
30 unsigned short * lcd_init(unsigned char *parlcd_mem_base);
31
32
41 void lcd_draw_pixel(unsigned short *fb,int x, int y, unsigned short color);
```

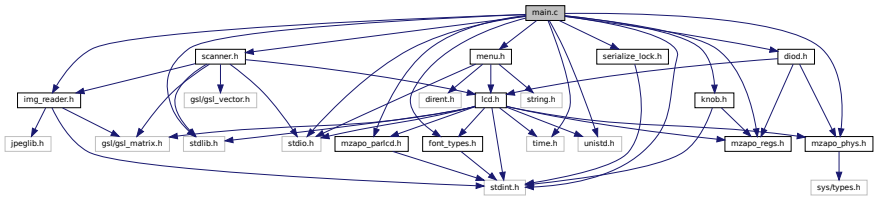
```
42
54 void lcd_draw_char(unsigned short *fb, int x, int y, font_descriptor_t *fdes, char ch, unsigned short
       color, unsigned int font_size);
62 int lcd_char_width(char ch, font_descriptor_t *fdes);
63
73 void lcd_draw_scaled_pixel_block(unsigned short *fb, int x, int y, unsigned short color, unsigned int
       scale);
86 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);
98 void lcd_draw_text(unsigned short *fb, int x, int y, font_descriptor_t *fdes, char *text, unsigned short
       color, unsigned int font_size);
99
106 void lcd_update_display(unsigned short * fb,unsigned char *parlcd_mem_base);
107
114 unsigned short lcd_grey(uint8_t intensity);
115
124 unsigned short lcd_color(uint8_t red, uint8_t green, uint8_t blue);
125
126
135 void lcd_draw_image(unsigned short * fb, int width, int height, gsl_matrix * image);
144 void lcd_draw_plus(unsigned short * fb, int x, int y, unsigned short color);
151 void lcd_fill_screen(unsigned short * fb, unsigned short color);
152
153 #endif // LCD_H
```

5.14 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:



Macros

- #define **_POSIX_C_SOURCE** 200112L

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.

5.14.1 Detailed Description

Main file for the MicroZed based MZ_APO board project.

5.14.2 Function Documentation

5.14.2.1 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

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

5.14.2.2 initialize_display()

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

Initialize the display.

Parameters

parlcd_mem_base	Base address of the LCD memory.

Returns

Pointer to the framebuffer.

5.14.2.3 initialize_memory()

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

Initialize memory mapping.

Parameters

phys_base	The physical base address.
size	The size of the memory region.

Returns

Pointer to the mapped memory.

5.14.2.4 load_image()

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

Load an image from a file.

Parameters

file_path	The path to the image file.
gray_image	Pointer to the grayscale image matrix.
width	Pointer to the width of the image.
height	Pointer to the height of the image.

5.14.2.5 main()

```
int main (
            void  )
```

Main function.

Returns

Exit status.

5.14.2.6 save_transformed_image()

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

Save the transformed image to a file.

Parameters

image	The transformed image matrix.
filename	The name of the output file.

5.14.2.7 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.

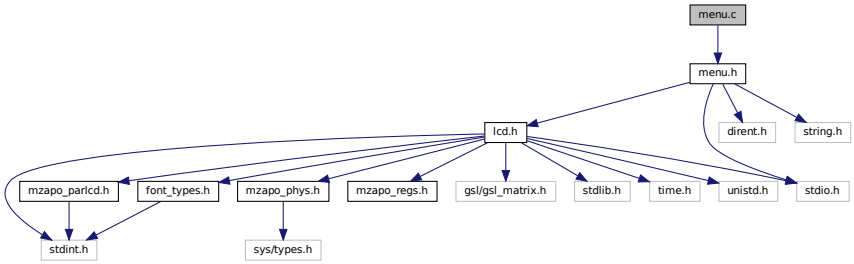
Parameters

src_mat	Source matrix of points.
dst_mat	Destination matrix of points.
H	Homography matrix.
width	The width of the image.
height	The height of the image.

5.15 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.

5.15.1 Detailed Description

Implementation of menu functions.

5.15.2 Function Documentation

5.15.2.1 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

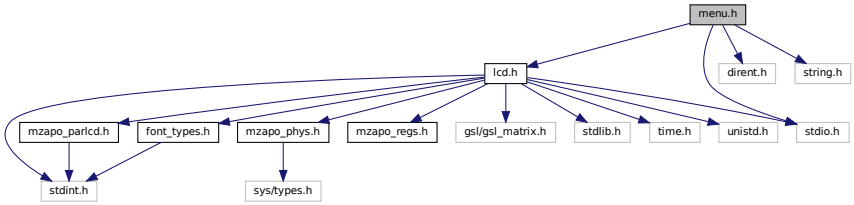
file_name	Buffer to store the file name.
file_number	The number of the file in the list.

5.16 menu.h File Reference

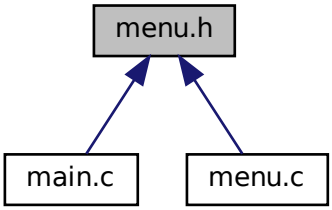
Header file for menu functions.

```
#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:



Macros

- #define **APP_DIR** "/tmp/filkiana/"

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.

5.16.1 Detailed Description

Header file for menu functions.

5.16.2 Function Documentation

5.16.2.1 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

file_name	Buffer to store the file name.
file_number	The number of the file in the list.

5.17 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
```

5.18 mzapo_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 mzapo_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*/
```

5.19 mzapo_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 mzapo_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*/
```

5.20 mzapo_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 mzapo_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
```

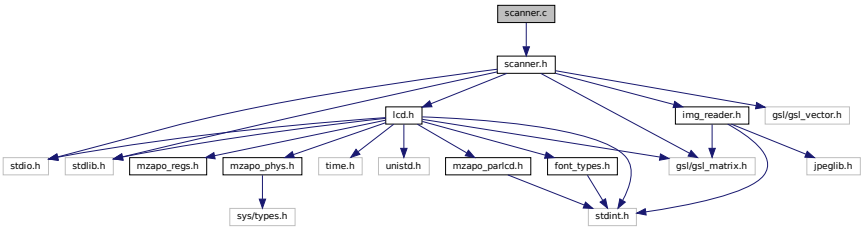
```
17
18 /*
19 Complete description of the educational MZ_APO design registers
20 can be found at
21
22 https://cw.fel.cvut.cz/wiki/courses/b35apo/documentation/mz_apo/start
23
24 The peripherals VHDL sources can be found in the repository
25
26 http://rtime.felk.cvut.cz/gitweb/fpga/zynq/canbench-sw.git/tree/refs/heads/microzed_apo:/system/ip
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*/
```

5.21 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.

5.21.1 Detailed Description

Implementation of matrix and transformation functions.

5.21.2 Function Documentation

5.21.2.1 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

gray_image	The source grayscale image matrix.
H	The homography matrix.
image_wrapped	The destination image matrix.
width	The width of the source image.
height	The height of the source image.

5.21.2.2 compute_perspective_transform()

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

Compute the perspective transform matrix.

Parameters

src	The source points matrix.
dst	The destination points matrix.
H	The homography matrix.

5.21.2.3 print_matrix()

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

Print a matrix to the console.

Parameters

m	The matrix to print.
name	The name of the matrix.

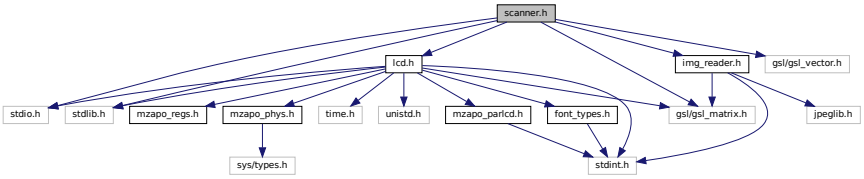
5.22 scanner.h File Reference

Header file for matrix and transformation functions.

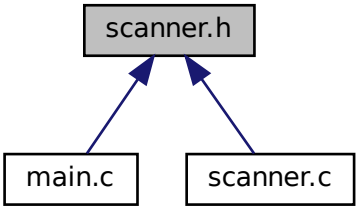
```
#include <stdio.h>
#include <stdlib.h>
```

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

Include dependency graph for scanner.h:



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



Functions

- import< gsl/gsl_linalg.h > import< gsl/gsl_blas.h > 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.

5.22.1 Detailed Description

Header file for matrix and transformation functions.

5.22.2 Function Documentation

5.22.2.1 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

gray_image	The source grayscale image matrix.
H	The homography matrix.
image_wrapped	The destination image matrix.
width	The width of the source image.
height	The height of the source image.

5.22.2.2 compute_perspective_transform()

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

Compute the perspective transform matrix.

Parameters

src	The source points matrix.
dst	The destination points matrix.
H	The homography matrix.

5.22.2.3 print_matrix()

```
import< gsl/gsl_linalg.h > import< gsl/gsl_blas.h > void print_matrix (
            const gsl_matrix * m,
            const char * name )
```

Print a matrix to the console.

Parameters

m	The matrix to print.
name	The name of the matrix.

5.23 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 import <gsl/gsl_linalg.h>
17 import <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 */
```

5.24 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