lcdlib-st7565p

Generated by Doxygen 1.8.13

Contents

1 lcdlib-st7565p **1**

2 Data Structure Index **3**

2.1 Data Structures 3

3 File Index **5**

3.1 File List 5

4 Data Structure Documentation **7**

4.1 LCD_BITMAP Struct Reference 7

4.1.1 Detailed Description 7

4.1.2 Field Documentation 7

4.1.2.1 bpc 8

4.1.2.2 data 8

4.1.2.3 height 8

4.1.2.4 size 8

4.1.2.5 width 8

4.2 LCD_GRAPHICS Struct Reference 8

4.2.1 Detailed Description 9

4.2.2 Field Documentation 9

4.2.2.1 dirty 9

4.2.2.2 dirty_x0 9

4.2.2.3 dirty_x1 9

4.2.2.4 dirty_y0 10

4.2.2.5 dirty_y1 10

4.2.2.6 graphicsmode 10

5 File Documentation 11

5.1 bitmap.c File Reference 11

5.1.1 Function Documentation 11

5.1.1.1 lcd_bitmapalloc() 12

5.1.1.2 lcd_bitmapclear() 12

5.1.1.3 lcd_bitmapfree() 12

5.1.1.4 lcd_getbitmapmemusage() 13

5.1.1.5 lcd_settotalbitmapmemlimit() 13

5.1.2 Variable Documentation 13

5.1.2.1 _lcd_totalmemlimit 13

5.1.2.2 _lcd_totalmemory 13

5.2 bitmap.h File Reference 13

5.2.1 Function Documentation 14

5.2.1.1 lcd_bitmapalloc() 14

5.2.1.2 lcd_bitmapclear() 14

5.2.1.3 lcd_bitmapfree() 15

5.2.1.4 lcd_getbitmapmemusage() 15

5.2.1.5 lcd_settotalbitmapmemlimit() 15

5.3 callbacks.h File Reference 16

5.3.1 Typedef Documentation 16

5.3.1.1 lcd_delay_callback 16

5.3.1.2 lcd_log_callback 16

5.3.1.3 lcd_pin_callback 17

5.3.1.4 lcd_spi_callback 17

5.4 charset.c File Reference 17

5.4.1 Variable Documentation 17

5.4.1.1 LCD_MEMORY 17

5.5 charset.h File Reference 17

5.5.1 Variable Documentation 18

5.5.1.1 LCD_MEMORY 18

5.6 graphics.c File Reference 18
5.6.1 Macro Definition Documentation 19
5.6.1.1 MAX 19
5.6.1.2 MIN 19
5.6.2 Function Documentation 19
5.6.2.1 _lcd_drawbyte() 19
5.6.2.2 _lcd_drawhalfchar() 19
5.6.2.3 _lcd_drawhalfline() 20
5.6.2.4 _lcd_drawpixel() 20
5.6.2.5 _lcd_drawpoint() 20
5.6.2.6 _lcd_fillline() 20
5.6.2.7 _lcd_refreshdirtyrect() 20
5.6.2.8 lcd_drawcircle() 21
5.6.2.9 lcd_drawline() 21
5.6.2.10 lcd_drawpoint() 21
5.6.2.11 lcd_drawtext() 22
5.6.2.12 lcd_fillrect() 22
5.7 graphics.h File Reference 23
5.7.1 Enumeration Type Documentation 23
5.7.1.1 lcd_graphicsmode 23
5.7.2 Function Documentation 24
5.7.2.1 lcd_drawcircle() 24
5.7.2.2 lcd_drawline() 24
5.7.2.3 lcd_drawpoint() 25
5.7.2.4 lcd_drawtext() 25
5.7.2.5 lcd_fillrect() 26
5.8 lcd.c File Reference 26
5.8.1 Function Documentation 27
5.8.1.1 _lcd_char() 27
5.8.1.2 _lcd_charwidth() 27

5.8.1.3 _lcd_data() 27
5.8.1.4 _lcd_linewidth() 28
5.8.1.5 lcd_clear() 28
5.8.1.6 lcd_command() 28
5.8.1.7 lcd_init() 28
5.8.1.8 lcd_printbitmap() 29
5.8.1.9 lcd_printgraphicbitmap() 29
5.8.1.10 lcd_printtext() 29
5.8.1.11 lcd_setlogfunction() 30
5.8.1.12 lcd_textsize() 30
5.8.2 Variable Documentation 31
5.8.2.1 _lcd_a0_set 31
5.8.2.2 _lcd_height 31
5.8.2.3 _lcd_reset_set 31
5.8.2.4 _lcd_select_set 31
5.8.2.5 _lcd_spi 31
5.8.2.6 _lcd_width 31
5.9 lcd.h File Reference 31
5.9.1 Enumeration Type Documentation 32
5.9.1.1 anonymous enum 32
5.9.1.2 lcd_commands 33
5.9.2 Function Documentation 33
5.9.2.1 lcd_clear() 33
5.9.2.2 lcd_command() 33
5.9.2.3 lcd_init() 33
5.9.2.4 lcd_printbitmap() 34
5.9.2.5 lcd_printgraphicbitmap() 34
5.9.2.6 lcd_printtext() 36
5.9.2.7 lcd_setlogfunction() 36
5.9.2.8 lcd_textsize() 36

5.10 log.c File Reference 37
5.10.1 Function Documentation 37
5.10.1.1 lcd_logchar() 37
5.10.1.2 lcd_logint() 37
5.10.1.3 lcd_logstr() 38
5.10.1.4 lcd_ramcheck() 38
5.10.2 Variable Documentation 38
5.10.2.1 _lcd_log 38
5.11 log.h File Reference 38
5.11.1 Function Documentation 38
5.11.1.1 lcd_logchar() 39
5.11.1.2 lcd_logint() 39
5.11.1.3 lcd_logstr() 39
5.11.1.4 lcd_ramcheck() 39
5.11.2 Variable Documentation 39
5.11.2.1 _lcd_log 39
5.12 README.md File Reference 40
5.13 Release/bitmap.d File Reference 40
5.14 Release/charset.d File Reference 40
5.15 Release/graphics.d File Reference 40
5.16 Release/lcd.d File Reference 40
5.17 Release/log.d File Reference 40
5.18 settings.h File Reference 40
5.18.1 Macro Definition Documentation 40
5.18.1.1 LCD_ARCH_AVR 40
5.18.1.2 LCD_LOGGING 40
5.18.1.3 LCD_MEM_READ 40
5.18.1.4 LCD_MEMORY 40

Index **41**

Chapter 1

lcdlib-st7565p

The lcdlib-st7565p is a C library for microcontroller interface to a write-only serial connection to a ST7565P LCD controller.

The ST7565P LCD controller does not provide a read mode when connected via a serial interface. As one has to set eight bits at once while writing to the display it is difficult to do graphical operations without read access. This library buffers the display content of specified regions or the whole display in the RAM. For each pixel in these regions one bit of RAM is needed. So for a 128x64 pixel region 8192 bits or 1kbyte of RAM is needed. While beeing very memory intensive this method allows to do graphic manipulations and drawing without overwriting previous pixels. For the display areas outside of the specified graphic regions only text can be displayed and must be aligned to text lines with a height of eight pixels.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

LCD_BITMAP
Bitmap data structure 7
LCD_GRAPHICS
Graphics info structure 8

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

bitmap.c 11
bitmap.h 13
callbacks.h 16
charset.c 17
charset.h 17
graphics.c 18
graphics.h 23
lcd.c 26
lcd.h 31
log.c 37
log.h 38
settings.h 40
Release/bitmap.d 40
Release/charset.d 40
Release/graphics.d 40
Release/lcd.d 40
Release/log.d 40

Chapter 4

Data Structure Documentation

4.1 LCD_BITMAP Struct Reference

Bitmap data structure.

```
#include <bitmap.h>
```

Data Fields

- uint8_t width

  Width of the bitmap in pixels.
- uint8_t height

  Height of the bitmap in pixels.
- uint16_t size

  Size of the data array in bytes.
- uint8_t bpc

  Bytes per pixel column.
- uint8_t ∗ data

  The bitmap pixel data storage array.

4.1.1 Detailed Description

Bitmap data structure.

A bitmap contains graphical data. Each bit in the data field represents one pixel. This structure contains information aboout the bitmap, such as width and height in pixels, the byte-size of the data array and the data array itself.

4.1.2 Field Documentation

4.1.2.1 bpc

```
uint8_t bpc
```

Bytes per pixel column.

Number of bytes to store all the pixels of one bitmap column (y-direction).

4.1.2.2 data

```
uint8_t* data
```

The bitmap pixel data storage array.

Each byte represents eight pixels in y-direction. LSB is lower y and MSB is higher y value. Successive bytes follow each other in x-direction. That way <width> bytes represent an area of <width> x 8 pixels. If (<height> mod 8) does not equal zero the upper bits of the last byte row are padded, but value is irrelevant.

4.1.2.3 height

```
uint8_t height
```

Height of the bitmap in pixels.

4.1.2.4 size

```
uint16_t size
```

Size of the data array in bytes.

4.1.2.5 width

```
uint8_t width
```

Width of the bitmap in pixels.

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

- bitmap.h

4.2 LCD_GRAPHICS Struct Reference

Graphics info structure.

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

Data Fields

- uint8_t graphicsmode

  Composition mode.
- bool dirty

  Bitmap needs to be reprinted.
- uint8_t dirty_x0

  Lower corner x of dirty rect.
- uint8_t dirty_y0

  Lower corner y of dirty rect.
- uint8_t dirty_x1

  Upper corner x of dirty rect.
- uint8_t dirty_y1

  Upper corner y of dirty rect.

4.2.1 Detailed Description

Graphics info structure.

This structure stores information about the graphics operations.

4.2.2 Field Documentation

4.2.2.1 dirty

```
bool dirty
```

Bitmap needs to be reprinted.

4.2.2.2 dirty_x0

```
uint8_t dirty_x0
```

Lower corner x of dirty rect.

4.2.2.3 dirty_x1

```
uint8_t dirty_x1
```

Upper corner x of dirty rect.

4.2.2.4 dirty_y0

```
uint8_t dirty_y0
```

Lower corner y of dirty rect.

4.2.2.5 dirty_y1

```
uint8_t dirty_y1
```

Upper corner y of dirty rect.

4.2.2.6 graphicsmode

```
uint8_t graphicsmode
```

Composition mode.

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

- graphics.h

Chapter 5

File Documentation

5.1 bitmap.c File Reference

```
#include "bitmap.h"
#include "stdlib.h"
#include "string.h"
```

Functions

- uint16_t lcd_getbitmapmemusage ()

  Get the amount of memory used by all allocated bitmaps.
- bool lcd_settotalbitmapmemlimit (uint16_t limit)

  Set the maximum amount of memory used by all allocated bitmaps.
- bool lcd_bitmapalloc (LCD_BITMAP ∗bmp, uint8_t width, uint8_t height)

  Allocate memory for a bitmap.
- void lcd_bitmapfree (LCD_BITMAP ∗bmp)

  Free memory of a bitmap.
- void lcd_bitmapclear (LCD_BITMAP ∗bmp, bool inv)

  Clear the contents of a bitmap.

Variables

- uint16_t _lcd_totalmemlimit = 0xFFFF
- uint16_t _lcd_totalmemory = 0

5.1.1 Function Documentation

5.1.1.1 lcd_bitmapalloc()

```
bool lcd_bitmapalloc (
            LCD_BITMAP * bmp,
            uint8_t width,
            uint8_t height )
```

Allocate memory for a bitmap.

Returns

If the memory used when trying to create this bitmap is higher than the set limit it returns false and will not be allocated.

Parameters

bmp	Pointer to uninitialized LCD_BITMAP structure to initialize.
width	Bitmap width in pixels.
height	Bitmap height in pixels.

5.1.1.2 lcd_bitmapclear()

```
void lcd_bitmapclear (
            LCD_BITMAP * bmp,
            bool inv )
```

Clear the contents of a bitmap.

Parameters

bmp	Pointer to initialized LCD_BITMAP structure.
inv	If true, set all pixels on, else set all pixels off.

5.1.1.3 lcd_bitmapfree()

```
void lcd_bitmapfree (
            LCD_BITMAP * bmp )
```

Free memory of a bitmap.

Parameters

bmp	Pointer to initialized LCD_BITMAP structure to free.

5.1.1.4 lcd_getbitmapmemusage()

```
uint16_t lcd_getbitmapmemusage ( )
```

Get the amount of memory used by all allocated bitmaps.

Returns

The amount of memory used by all allocated bitmaps in bytes.

5.1.1.5 lcd_settotalbitmapmemlimit()

```
bool lcd_settotalbitmapmemlimit (
             uint16_t limit )
```

Set the maximum amount of memory used by all allocated bitmaps.

Returns

When more memory than `limit` is currently allocated the return value is false and the new limit is not set.

Parameters

limit	The memory limit in bytes.

5.1.2 Variable Documentation

5.1.2.1 _lcd_totalmemlimit

```
uint16_t _lcd_totalmemlimit = 0xFFFF
```

5.1.2.2 _lcd_totalmemory

```
uint16_t _lcd_totalmemory = 0
```

5.2 bitmap.h File Reference

```
#include "stdint.h"
#include "stdbool.h"
#include "settings.h"
```

Data Structures

- struct LCD_BITMAP

  Bitmap data structure.

Functions

- uint16_t lcd_getbitmapmemusage ()

  Get the amount of memory used by all allocated bitmaps.
- bool lcd_settotalbitmapmemlimit (uint16_t limit)

  Set the maximum amount of memory used by all allocated bitmaps.
- bool lcd_bitmapalloc (LCD_BITMAP ∗bmp, uint8_t width, uint8_t height)

  Allocate memory for a bitmap.
- void lcd_bitmapfree (LCD_BITMAP ∗bmp)

  Free memory of a bitmap.
- void lcd_bitmapclear (LCD_BITMAP ∗bmp, bool inv)

  Clear the contents of a bitmap.

5.2.1 Function Documentation

5.2.1.1 lcd_bitmapalloc()

```
bool lcd_bitmapalloc (
            LCD_BITMAP * bmp,
            uint8_t width,
            uint8_t height )
```

Allocate memory for a bitmap.

Returns

If the memory used when trying to create this bitmap is higher than the set limit it returns false and will not be allocated.

Parameters

bmp	Pointer to uninitialized LCD_BITMAP structure to initialize.
width	Bitmap width in pixels.
height	Bitmap height in pixels.

5.2.1.2 lcd_bitmapclear()

```
void lcd_bitmapclear (
            LCD_BITMAP * bmp,
            bool inv )
```

Clear the contents of a bitmap.

Parameters

bmp	Pointer to initialized LCD_BITMAP structure.
inv	If true, set all pixels on, else set all pixels off.

5.2.1.3 lcd_bitmapfree()

```
void lcd_bitmapfree (
            LCD_BITMAP * bmp )
```

Free memory of a bitmap.

Parameters

bmp	Pointer to initialized LCD_BITMAP structure to free.

5.2.1.4 lcd_getbitmapmemusage()

```
uint16_t lcd_getbitmapmemusage ( )
```

Get the amount of memory used by all allocated bitmaps.

Returns

The amount of memory used by all allocated bitmaps in bytes.

5.2.1.5 lcd_settotalbitmapmemlimit()

```
bool lcd_settotalbitmapmemlimit (
            uint16_t limit )
```

Set the maximum amount of memory used by all allocated bitmaps.

Returns

When more memory than `limit` is currently allocated the return value is false and the new limit is not set.

Parameters

limit	The memory limit in bytes.

5.3 callbacks.h File Reference

```
#include "stdint.h"
#include "settings.h"
```

Typedefs

- typedef void(∗ lcd_delay_callback) (int ms)

  Callback funtion pointer type for a delay function with integer time parameter in milliseconds.
- typedef void(∗ lcd_pin_callback) (uint8_t state)

  Callback funtion pointer type for a pin set function with parameter for the new pin state (0 = off, 1 = on).
- typedef uint8_t(∗ lcd_spi_callback) (uint8_t data)

  Callback funtion pointer type for a spi transceive function with parameter for the data to send. Returns the received data.
- typedef void(∗ lcd_log_callback) (char ∗str)

  Callback funtion pointer type for a logging function with parameter for the string to log.

5.3.1 Typedef Documentation

5.3.1.1 lcd_delay_callback

```
typedef void(* lcd_delay_callback) (int ms)
```

Callback funtion pointer type for a delay function with integer time parameter in milliseconds.

5.3.1.2 lcd_log_callback

```
typedef void(* lcd_log_callback) (char *str)
```

Callback funtion pointer type for a logging function with parameter for the string to log.

5.3.1.3 lcd_pin_callback

```
typedef void(* lcd_pin_callback) (uint8_t state)
```

Callback funtion pointer type for a pin set function with parameter for the new pin state (0 = off, 1 = on).

5.3.1.4 lcd_spi_callback

```
typedef uint8_t(* lcd_spi_callback) (uint8_t data)
```

Callback funtion pointer type for a spi transceive function with parameter for the data to send. Returns the received data.

5.4 charset.c File Reference

```
#include "charset.h"
```

Variables

- const uint8_t lcd_charunknown [ ] LCD_MEMORY = { 0x7F, 0x41, 0x41, 0x41, 0x7F }

  Fixed storage for unknown char data.

5.4.1 Variable Documentation

5.4.1.1 LCD_MEMORY

```
const uint8_t lcd_charset [ ] LCD_MEMORY = { 0x7F, 0x41, 0x41, 0x41, 0x7F }
```

Fixed storage for unknown char data.

Fixed storage for the character set data.

5.5 charset.h File Reference

```
#include "settings.h"
```

Variables

- const uint8_t lcd_charunknown [ ] LCD_MEMORY

  Fixed storage for unknown char data.

5.5.1 Variable Documentation

5.5.1.1 LCD_MEMORY

```
const uint8_t lcd_charset [ ] LCD_MEMORY
```

Fixed storage for unknown char data.

Fixed storage for the character set data.

5.6 graphics.c File Reference

```
#include "graphics.h"
#include "charset.h"
#include "log.h"
#include "stdbool.h"
#include "stdlib.h"
#include "string.h"
```

Macros

- #define MAX(a, b) ((a < b) ? b : a)
- #define MIN(a, b) ((a < b) ? a : b)

Functions

- void _lcd_drawpixel (LCD_BITMAP ∗canvas, uint16_t addr, uint8_t bit, uint8_t mode)
- void _lcd_refreshdirtyrect (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x0, int y0, int x1, int y1)
- void _lcd_drawpoint (LCD_BITMAP ∗canvas, int x, int y, uint8_t mode)
- void lcd_drawpoint (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x, int y)

  Draw a point to a bitmap canvas.
- void lcd_drawline (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x0, int y0, int x1, int y1)

  Draw a line to a bitmap canvas.
- void lcd_drawcircle (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x0, int y0, int radius)

  Draw a circle to a bitmap canvas.
- void _lcd_drawbyte (uint8_t ∗addr, uint8_t value, uint8_t mode)
- void _lcd_fillline (uint8_t ∗startaddr, uint8_t dx, uint8_t pattern, uint8_t mode)
- void lcd_fillrect (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x0, int y0, int x1, int y1)

  Draw a filled rectangle to a bitmap canvas.
- void _lcd_drawhalfchar (LCD_BITMAP ∗canvas, int ∗x, uint8_t ∗lineaddr, int8_t shift_bits, char c, uint8_t mode, bool space)
- char ∗ _lcd_drawhalfline (LCD_BITMAP ∗canvas, int x, uint8_t ∗lineaddr, uint8_t shift_bits, char ∗txt, uint8_t mode, int ∗maxx)
- void lcd_drawtext (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x, int y, char ∗txt)

  Draw text to a bitmap canvas.

5.6.1 Macro Definition Documentation

5.6.1.1 MAX

```
#define MAX(
              a,
              b ) ((a < b) ? b : a)
```

5.6.1.2 MIN

```
#define MIN(
              a,
              b ) ((a < b) ? a : b)
```

5.6.2 Function Documentation

5.6.2.1 _lcd_drawbyte()

```
void _lcd_drawbyte (
             uint8_t * addr,
             uint8_t value,
             uint8_t mode )
```

5.6.2.2 _lcd_drawhalfchar()

```
void _lcd_drawhalfchar (
             LCD_BITMAP * canvas,
             int * x,
             uint8_t * lineaddr,
             int8_t shift_bits,
             char c,
             uint8_t mode,
             bool space )
```

5.6.2.3 _lcd_drawhalfline()

```
char* _lcd_drawhalfline (
             LCD_BITMAP * canvas,
             int x,
             uint8_t * lineaddr,
             uint8_t shift_bits,
             char * txt,
             uint8_t mode,
             int * maxx )
```

5.6.2.4 _lcd_drawpixel()

```
void _lcd_drawpixel (
             LCD_BITMAP * canvas,
             uint16_t addr,
             uint8_t bit,
             uint8_t mode )
```

5.6.2.5 _lcd_drawpoint()

```
void _lcd_drawpoint (
             LCD_BITMAP * canvas,
             int x,
             int y,
             uint8_t mode )
```

5.6.2.6 _lcd_fillline()

```
void _lcd_fillline (
             uint8_t * startaddr,
             uint8_t dx,
             uint8_t pattern,
             uint8_t mode )
```

5.6.2.7 _lcd_refreshdirtyrect()

```
void _lcd_refreshdirtyrect (
             LCD_BITMAP * canvas,
             LCD_GRAPHICS * mode,
             int x0,
             int y0,
             int x1,
             int y1 )
```

5.6.2.8 lcd_drawcircle()

```
void lcd_drawcircle (
            LCD_BITMAP * canvas,
            LCD_GRAPHICS * mode,
            int x0,
            int y0,
            int radius )
```

Draw a circle to a bitmap canvas.

Implementation taken from wikipedia (https://de.wikipedia.org/wiki/Bresenham-Algorithmus).

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x0	Center point on x-axis.
y0	Center point on y-axis.
radius	Circle radius in pixels.
mode	Composition mode of the drawing operation.

5.6.2.9 lcd_drawline()

```
void lcd_drawline (
            LCD_BITMAP * canvas,
            LCD_GRAPHICS * mode,
            int x0,
            int y0,
            int x1,
            int y1 )
```

Draw a line to a bitmap canvas.

Implementation taken from wikipedia (https://de.wikipedia.org/wiki/Bresenham-Algorithmus).

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x0	Start point on x-axis.
y0	Start point on y-axis.
x1	End point on x-axis.
y2	End point on y-axis.
mode	Composition mode of the drawing operation.

5.6.2.10 lcd_drawpoint()

```
void lcd_drawpoint (
            LCD_BITMAP * canvas,
            LCD_GRAPHICS * mode,
            int x,
            int y )
```

Draw a point to a bitmap canvas.

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x	Position of the point in x-direction in pixels.
y	Position of the point in y-direction in pixels.
mode	Composition mode of the drawing operation.

5.6.2.11 lcd_drawtext()

```
void lcd_drawtext (
            LCD_BITMAP * canvas,
            LCD_GRAPHICS * mode,
            int x,
            int y,
            char * txt )
```

Draw text to a bitmap canvas.

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x	Starting point of the text on the x-axis.
y	Starting point of the text on the y-axis.
txt	The string to draw.
mode	Composition mode of the drawing operation.

5.6.2.12 lcd_fillrect()

```
void lcd_fillrect (
            LCD_BITMAP * canvas,
            LCD_GRAPHICS * mode,
            int x0,
            int y0,
            int x1,
            int y1 )
```

Draw a filled rectangle to a bitmap canvas.

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x0	Lower value x-axis point.
y0	Lower value y-axis point.
x1	Higher value x-axis point.
y2	Higher value y-axis point.
mode	Composition mode of the drawing operation.

5.7 graphics.h File Reference

```
#include "stdint.h"
#include "stdbool.h"
#include "bitmap.h"
#include "settings.h"
```

Data Structures

- struct LCD_GRAPHICS

  Graphics info structure.

Enumerations

- enum lcd_graphicsmode { LCD_SETPOINT = 0b01, LCD_CLEARPOINT = 0b10, LCD_INVERTPOINT = LCD_SETPOINT | LCD_CLEARPOINT, LCD_THICKPOINT = 0b100 }

  Composition mode for drawings onto a bitmap graphics.

Functions

- void lcd_drawpoint (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x, int y)

  Draw a point to a bitmap canvas.
- void lcd_drawline (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x0, int y0, int x1, int y1)

  Draw a line to a bitmap canvas.
- void lcd_drawcircle (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x0, int y0, int radius)

  Draw a circle to a bitmap canvas.
- void lcd_fillrect (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x0, int y0, int x1, int y1)

  Draw a filled rectangle to a bitmap canvas.
- void lcd_drawtext (LCD_BITMAP ∗canvas, LCD_GRAPHICS ∗mode, int x, int y, char ∗txt)

  Draw text to a bitmap canvas.

5.7.1 Enumeration Type Documentation

5.7.1.1 lcd_graphicsmode

```
enum lcd_graphicsmode
```

Composition mode for drawings onto a bitmap graphics.

Enumerator

LCD_SETPOINT	Set the pixels on.
LCD_CLEARPOINT	Set the pixels off.
LCD_INVERTPOINT	invert the pixels
LCD_THICKPOINT	Draw thick pixels.

5.7.2 Function Documentation

5.7.2.1 lcd_drawcircle()

```
void lcd_drawcircle (
             LCD_BITMAP * canvas,
             LCD_GRAPHICS * mode,
             int x0,
             int y0,
             int radius )
```

Draw a circle to a bitmap canvas.

Implementation taken from wikipedia (`https://de.wikipedia.org/wiki/Bresenham-Algorithmus`).

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x0	Center point on x-axis.
y0	Center point on y-axis.
radius	Circle radius in pixels.
mode	Composition mode of the drawing operation.

5.7.2.2 lcd_drawline()

```
void lcd_drawline (
             LCD_BITMAP * canvas,
             LCD_GRAPHICS * mode,
             int x0,
             int y0,
             int x1,
             int y1 )
```

Draw a line to a bitmap canvas.

Implementation taken from wikipedia (`https://de.wikipedia.org/wiki/Bresenham-Algorithmus`).

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x0	Start point on x-axis.
y0	Start point on y-axis.
x1	End point on x-axis.
y2	End point on y-axis.
mode	Composition mode of the drawing operation.

5.7.2.3 lcd_drawpoint()

```
void lcd_drawpoint (
            LCD_BITMAP * canvas,
            LCD_GRAPHICS * mode,
            int x,
            int y )
```

Draw a point to a bitmap canvas.

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x	Position of the point in x-direction in pixels.
y	Position of the point in y-direction in pixels.
mode	Composition mode of the drawing operation.

5.7.2.4 lcd_drawtext()

```
void lcd_drawtext (
            LCD_BITMAP * canvas,
            LCD_GRAPHICS * mode,
            int x,
            int y,
            char * txt )
```

Draw text to a bitmap canvas.

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x	Starting point of the text on the x-axis.
y	Starting point of the text on the y-axis.
txt	The string to draw.
mode	Composition mode of the drawing operation.

5.7.2.5 lcd_fillrect()

```
void lcd_fillrect (
            LCD_BITMAP * canvas,
            LCD_GRAPHICS * mode,
            int x0,
            int y0,
            int x1,
            int y1 )
```

Draw a filled rectangle to a bitmap canvas.

Parameters

canvas	Pointer to initialized LCD_BITMAP structure of the canvas to draw on.
x0	Lower value x-axis point.
y0	Lower value y-axis point.
x1	Higher value x-axis point.
y2	Higher value y-axis point.
mode	Composition mode of the drawing operation.

5.8 lcd.c File Reference

```
#include "lcd.h"
#include "charset.h"
#include "log.h"
#include "string.h"
#include "stdlib.h"
```

Functions

- void _lcd_data (uint8_t data)
- void lcd_init (lcd_spi_callback spi_callback, lcd_delay_callback delay_callback, lcd_pin_callback reset_←
callback, lcd_pin_callback a0_callback, lcd_pin_callback select_callback, uint8_t lcd_width, uint8_t lcd_←
height)

  Initialize the lcd library.
- void lcd_setlogfunction (lcd_log_callback log_callback)

  Set the callback function pointer for the logging function.
- void lcd_command (uint8_t cmd)

  Send a command to the display.
- void _lcd_char (char c, bool inv, bool space)
- void lcd_printtext (uint8_t x, uint8_t line, char ∗str, bool inv)

  Set a text to the display.
- uint8_t _lcd_charwidth (char c)
- char ∗ _lcd_linewidth (char ∗str, uint8_t ∗len)

- void lcd_textsize (char ∗str, uint8_t ∗width, uint8_t ∗lines)

    Calculate the size of a text.
- void lcd_printbitmap (uint8_t x, uint8_t line, LCD_BITMAP ∗bmp)

    Print a bitmap to the display.
- void lcd_printgraphicbitmap (uint8_t x, uint8_t line, LCD_BITMAP ∗bmp, LCD_GRAPHICS ∗g)

    Print a bitmap to the display with the information about the dirty region.
- void lcd_clear ()

    Clear the bitmap contents (set all pixels off).

Variables

- lcd_spi_callback _lcd_spi
- lcd_pin_callback _lcd_reset_set
- lcd_pin_callback _lcd_a0_set
- lcd_pin_callback _lcd_select_set
- uint8_t _lcd_width
- uint8_t _lcd_height

5.8.1 Function Documentation

5.8.1.1 _lcd_char()

```
void _lcd_char (
            char c,
            bool inv,
            bool space )
```

5.8.1.2 _lcd_charwidth()

```
uint8_t _lcd_charwidth (
            char c )
```

5.8.1.3 _lcd_data()

```
void _lcd_data (
            uint8_t data )
```

5.8.1.4 _lcd_linewidth()

```
char* _lcd_linewidth (
              char * str,
              uint8_t * len )
```

5.8.1.5 lcd_clear()

```
void lcd_clear ( )
```

Clear the bitmap contents (set all pixels off).

5.8.1.6 lcd_command()

```
void lcd_command (
              uint8_t cmd )
```

Send a command to the display.

Parameters

cmd	The command to send (see lcd_commands).

5.8.1.7 lcd_init()

```
void lcd_init (
              lcd_spi_callback spi_callback,
              lcd_delay_callback delay_callback,
              lcd_pin_callback reset_callback,
              lcd_pin_callback a0_callback,
              lcd_pin_callback select_callback,
              uint8_t lcd_width,
              uint8_t lcd_height )
```

Initialize the lcd library.

Parameters

spi_callback	Callback function pointer to transceive data on the SPI peripheral.
delay_callback	Callback function pointer for a delay function (it is only used by this initialization method).
reset_callback	Callback function pointer to control the reset pin of the display.
a0_callback	Callback function pointer to control the a0 pin of the display.
select_callback	Callback function pointer to control the chip select pin of the display.
lcd_width	Width of the display in pixels.
lcd_height	Height of the display in pixels.

5.8.1.8 lcd_printbitmap()

```
void lcd_printbitmap (
            uint8_t x,
            uint8_t line,
            LCD_BITMAP * bmp )
```

Print a bitmap to the display.

Like text, bitmaps can also only be placed in the eight bit line grid and overwrite the existing content in the affected parts of the line. Ideally the height of a bitmap is a multiple of eight, so no pixels are wasted.

Parameters

x	Starting point of the bitmap on the x-axis in pixels.
line	Starting point of the bitmap on the y-axis in lines.
bmp	The bitmap to print.

5.8.1.9 lcd_printgraphicbitmap()

```
void lcd_printgraphicbitmap (
            uint8_t x,
            uint8_t line,
            LCD_BITMAP * bmp,
            LCD_GRAPHICS * g )
```

Print a bitmap to the display with the information about the dirty region.

Like text, bitmaps can also only be placed in the eight bit line grid and overwrite the existing content in the affected parts of the line. Ideally the height of a bitmap is a multiple of eight, so no pixels are wasted.

Parameters

x	Starting point of the bitmap on the x-axis in pixels.
line	Starting point of the bitmap on the y-axis in lines.
bmp	The bitmap to print.
g	Graphics info structure to reprint only the dirty parts.

5.8.1.10 lcd_printtext()

```
void lcd_printtext (
            uint8_t x,
            uint8_t line,
            char * str,
            bool inv )
```

Set a text to the display.

The text can be placed independently of allocated bitmaps but has to be aligned to the eight pixel line grid in y direction. Therefore the `line` Parameter is already in number of lines and not in pixels. Also the text will overwrite anything that it is printed over. To merge text into an image and use composition modes, draw it onto a bitmap.

Parameters

x	The start position of the text on the x-axis in pixels.
line	The start position of the text on the y-axis in lines.
str	The string to print.
inv	Inverts the pixel states of the text before printing.

5.8.1.11 lcd_setlogfunction()

```
void lcd_setlogfunction (
            lcd_log_callback log_callback )
```

Set the callback function pointer for the logging function.

Parameters

log_callback	Callback function pointer for the logging function (can be NULL).

5.8.1.12 lcd_textsize()

```
void lcd_textsize (
            char * str,
            uint8_t * width,
            uint8_t * lines )
```

Calculate the size of a text.

Parameters

str	The string to measure.
width	Pointer to the variable to be set by the function with the value of the pixel-with of the text.
lines	Pointer to the variable to be set by the function with the value of the line count of the text.

5.8.2 Variable Documentation

5.8.2.1 _lcd_a0_set

```
lcd_pin_callback _lcd_a0_set
```

5.8.2.2 _lcd_height

```
uint8_t _lcd_height
```

5.8.2.3 _lcd_reset_set

```
lcd_pin_callback _lcd_reset_set
```

5.8.2.4 _lcd_select_set

```
lcd_pin_callback _lcd_select_set
```

5.8.2.5 _lcd_spi

```
lcd_spi_callback _lcd_spi
```

5.8.2.6 _lcd_width

```
uint8_t _lcd_width
```

5.9 lcd.h File Reference

```
#include "stdbool.h"
#include "stdint.h"
#include "settings.h"
#include "bitmap.h"
#include "graphics.h"
#include "callbacks.h"
```

Enumerations

- enum { LCD_CMDON = 1, LCD_CMDOFF = 0 }

  Logical state of a command.
- enum lcd_commands {
  LCD_CMDDISPLAY = 0xAE, LCD_CMDSTARTLINESET = 0x40, LCD_CMDPAGEADDRSET = 0xB0, LCD_CMDCOLUMNADDRSETH = 0x10,
  LCD_CMDCOLUMNADDRSETL = 0x00, LCD_CMDCOLUMNREVERSE = 0xA0, LCD_CMDDISPLAYINVERSE = 0xA6, LCD_CMDSETALLPOINTS = 0xA4 }

  Commands that can be send to the display.

Functions

- void lcd_init (lcd_spi_callback spi_callback, lcd_delay_callback delay_callback, lcd_pin_callback reset_callback, lcd_pin_callback a0_callback, lcd_pin_callback select_callback, uint8_t lcd_width, uint8_t lcd_height)

  Initialize the lcd library.
- void lcd_setlogfunction (lcd_log_callback log_callback)

  Set the callback function pointer for the logging function.
- void lcd_command (uint8_t cmd)

  Send a command to the display.
- void lcd_printtext (uint8_t x, uint8_t line, char ∗str, bool inv)

  Set a text to the display.
- void lcd_textsize (char ∗str, uint8_t ∗width, uint8_t ∗lines)

  Calculate the size of a text.
- void lcd_printbitmap (uint8_t x, uint8_t line, LCD_BITMAP ∗bmp)

  Print a bitmap to the display.
- void lcd_printgraphicbitmap (uint8_t x, uint8_t line, LCD_BITMAP ∗bmp, LCD_GRAPHICS ∗g)

  Print a bitmap to the display with the information about the dirty region.
- void lcd_clear ()

  Clear the bitmap contents (set all pixels off).

5.9.1 Enumeration Type Documentation

5.9.1.1 anonymous enum

```
anonymous enum
```

Logical state of a command.

Enumerator

LCD_CMDON	Command ON.
LCD_CMDOFF	Command OFF.

5.9.1.2 lcd_commands

```
enum lcd_commands
```

Commands that can be send to the display.

Enumerator

LCD_CMDDISPLAY	Turn display on or off (OR with CMDON or CMDOFF).
LCD_CMDSTARTLINESET	Set start line address (OR with line number).
LCD_CMDPAGEADDRSET	Set page address to write to (OR with page number).
LCD_CMDCOLUMNADDRSETH	Set column address high nibble (OR with high nibble).
LCD_CMDCOLUMNADDRSETL	Set column address low nibble (OR with low nibble).
LCD_CMDCOLUMNREVERSE	Reverse column scan direction (OR with CMDON or CMDOFF).
LCD_CMDDISPLAYINVERSE	Inverse all display points (OR with CMDON or CMDOFF).
LCD_CMDSETALLPOINTS	Set all points ignoring data and inversion (OR with CMDON or CMDOFF).

5.9.2 Function Documentation

5.9.2.1 lcd_clear()

```
void lcd_clear ( )
```

Clear the bitmap contents (set all pixels off).

5.9.2.2 lcd_command()

```
void lcd_command (
            uint8_t cmd )
```

Send a command to the display.

Parameters

cmd	The command to send (see lcd_commands).

5.9.2.3 lcd_init()

```
void lcd_init (
            lcd_spi_callback spi_callback,
            lcd_delay_callback delay_callback,
            lcd_pin_callback reset_callback,
            lcd_pin_callback a0_callback,
            lcd_pin_callback select_callback,
            uint8_t lcd_width,
            uint8_t lcd_height )
```

Initialize the lcd library.

Parameters

spi_callback	Callback function pointer to transceive data on the SPI peripheral.
delay_callback	Callback function pointer for a delay function (it is only used by this initialization method).
reset_callback	Callback function pointer to control the reset pin of the display.
a0_callback	Callback function pointer to control the a0 pin of the display.
select_callback	Callback function pointer to control the chip select pin of the display.
lcd_width	Width of the display in pixels.
lcd_height	Height of the display in pixels.

5.9.2.4 lcd_printbitmap()

```
void lcd_printbitmap (
             uint8_t x,
             uint8_t line,
             LCD_BITMAP * bmp )
```

Print a bitmap to the display.

Like text, bitmaps can also only be placed in the eight bit line grid and overwrite the existing content in the affected parts of the line. Ideally the height of a bitmap is a multiple of eight, so no pixels are wasted.

Parameters

x	Starting point of the bitmap on the x-axis in pixels.
line	Starting point of the bitmap on the y-axis in lines.
bmp	The bitmap to print.

5.9.2.5 lcd_printgraphicbitmap()

```
void lcd_printgraphicbitmap (
             uint8_t x,
             uint8_t line,
             LCD_BITMAP * bmp,
             LCD_GRAPHICS * g )
```

Print a bitmap to the display with the information about the dirty region.

Like text, bitmaps can also only be placed in the eight bit line grid and overwrite the existing content in the affected parts of the line. Ideally the height of a bitmap is a multiple of eight, so no pixels are wasted.

Parameters

x	Starting point of the bitmap on the x-axis in pixels.
line	Starting point of the bitmap on the y-axis in lines.
bmp	The bitmap to print.
g	Graphics info structure to reprint only the dirty parts.

5.9.2.6 lcd_printtext()

```
void lcd_printtext (
            uint8_t x,
            uint8_t line,
            char * str,
            bool inv )
```

Set a text to the display.

The text can be placed independently of allocated bitmaps but has to be aligned to the eight pixel line grid in y direction. Therefore the `line` Parameter is already in number of lines and not in pixels. Also the text will overwrite anything that it is printed over. To merge text into an image and use composition modes, draw it onto a bitmap.

Parameters

x	The start position of the text on the x-axis in pixels.
line	The start position of the text on the y-axis in lines.
str	The string to print.
inv	Inverts the pixel states of the text before printing.

5.9.2.7 lcd_setlogfunction()

```
void lcd_setlogfunction (
            lcd_log_callback log_callback )
```

Set the callback function pointer for the logging function.

Parameters

log_callback	Callback function pointer for the logging function (can be NULL).

5.9.2.8 lcd_textsize()

```
void lcd_textsize (
            char * str,
            uint8_t * width,
            uint8_t * lines )
```

Calculate the size of a text.

Parameters

str	The string to measure.
width	Pointer to the variable to be set by the function with the value of the pixel-with of the text.
lines	Pointer to the variable to be set by the function with the value of the line count of the text.

5.10 log.c File Reference

```
#include "log.h"
#include "stdlib.h"
```

Functions

- void lcd_logstr (char ∗str, char ∗marker)
- void lcd_logchar (char c, char ∗marker)
- void lcd_logint (int i, char ∗marker)
- int lcd_ramcheck ()

Variables

- lcd_log_callback _lcd_log = 0

5.10.1 Function Documentation

5.10.1.1 lcd_logchar()

```
void lcd_logchar (
            char c,
            char * marker )
```

5.10.1.2 lcd_logint()

```
void lcd_logint (
            int i,
            char * marker )
```

5.10.1.3 lcd_logstr()

```
void lcd_logstr (
             char * str,
             char * marker )
```

5.10.1.4 lcd_ramcheck()

```
int lcd_ramcheck ( )
```

5.10.2 Variable Documentation

5.10.2.1 _lcd_log

```
lcd_log_callback _lcd_log = 0
```

5.11 log.h File Reference

```
#include "settings.h"
#include "callbacks.h"
```

Functions

- int lcd_ramcheck ()
- void lcd_logstr (char ∗str, char ∗marker)
- void lcd_logchar (char c, char ∗marker)
- void lcd_logint (int i, char ∗marker)

Variables

- lcd_log_callback _lcd_log

5.11.1 Function Documentation

5.11.1.1 lcd_logchar()

```
void lcd_logchar (
            char c,
            char * marker )
```

5.11.1.2 lcd_logint()

```
void lcd_logint (
            int i,
            char * marker )
```

5.11.1.3 lcd_logstr()

```
void lcd_logstr (
            char * str,
            char * marker )
```

5.11.1.4 lcd_ramcheck()

```
int lcd_ramcheck ( )
```

5.11.2 Variable Documentation

5.11.2.1 _lcd_log

```
lcd_log_callback _lcd_log
```

5.12 README.md File Reference

5.13 Release/bitmap.d File Reference

5.14 Release/charset.d File Reference

5.15 Release/graphics.d File Reference

5.16 Release/lcd.d File Reference

5.17 Release/log.d File Reference

5.18 settings.h File Reference

```
#include "avr/pgmspace.h"
#include "util/delay.h"
```

Macros

- #define LCD_ARCH_AVR
- #define LCD_LOGGING
- #define LCD_MEMORY PROGMEM
- #define LCD_MEM_READ(addr) pgm_read_byte(addr)

5.18.1 Macro Definition Documentation

5.18.1.1 LCD_ARCH_AVR

```
#define LCD_ARCH_AVR
```

5.18.1.2 LCD_LOGGING

```
#define LCD_LOGGING
```

5.18.1.3 LCD_MEM_READ

```
#define LCD_MEM_READ(
              addr ) pgm_read_byte(addr)
```

5.18.1.4 LCD_MEMORY

```
#define LCD_MEMORY PROGMEM
```

Index

_lcd_a0_set
  lcd.c, 31
_lcd_char
  lcd.c, 27
_lcd_charwidth
  lcd.c, 27
_lcd_data
  lcd.c, 27
_lcd_drawbyte
  graphics.c, 19
_lcd_drawhalfchar
  graphics.c, 19
_lcd_drawhalfline
  graphics.c, 19
_lcd_drawpixel
  graphics.c, 20
_lcd_drawpoint
  graphics.c, 20
_lcd_fillline
  graphics.c, 20
_lcd_height
  lcd.c, 31
_lcd_linewidth
  lcd.c, 27
_lcd_log
  log.c, 38
  log.h, 39
_lcd_refreshdirtyrect
  graphics.c, 20
_lcd_reset_set
  lcd.c, 31
_lcd_select_set
  lcd.c, 31
_lcd_spi
  lcd.c, 31
_lcd_totalmemlimit
  bitmap.c, 13
_lcd_totalmemory
  bitmap.c, 13
_lcd_width
  lcd.c, 31

bitmap.c, 11
  _lcd_totalmemlimit, 13
  _lcd_totalmemory, 13
  lcd_bitmapalloc, 11
  lcd_bitmapclear, 12
  lcd_bitmapfree, 12
  lcd_getbitmapmemusage, 12
  lcd_settotalbitmapmemlimit, 13
bitmap.h, 13
  lcd_bitmapalloc, 14
  lcd_bitmapclear, 14
  lcd_bitmapfree, 15
  lcd_getbitmapmemusage, 15
  lcd_settotalbitmapmemlimit, 15
bpc
  LCD_BITMAP, 7

callbacks.h, 16
  lcd_delay_callback, 16
  lcd_log_callback, 16
  lcd_pin_callback, 16
  lcd_spi_callback, 17
charset.c, 17
  LCD_MEMORY, 17
charset.h, 17
  LCD_MEMORY, 18

data
  LCD_BITMAP, 8
dirty
  LCD_GRAPHICS, 9
dirty_x0
  LCD_GRAPHICS, 9
dirty_x1
  LCD_GRAPHICS, 9
dirty_y0
  LCD_GRAPHICS, 9
dirty_y1
  LCD_GRAPHICS, 10

graphics.c, 18
  _lcd_drawbyte, 19
  _lcd_drawhalfchar, 19
  _lcd_drawhalfline, 19
  _lcd_drawpixel, 20
  _lcd_drawpoint, 20
  _lcd_fillline, 20
  _lcd_refreshdirtyrect, 20
  lcd_drawcircle, 20
  lcd_drawline, 21
  lcd_drawpoint, 21
  lcd_drawtext, 22
  lcd_fillrect, 22
  MAX, 19
  MIN, 19
graphics.h, 23
  lcd_drawcircle, 24
  lcd_drawline, 24

lcd_drawpoint, 25
lcd_drawtext, 25
lcd_fillrect, 26
lcd_graphicsmode, 23
graphicsmode
LCD_GRAPHICS, 10

height
LCD_BITMAP, 8

LCD_ARCH_AVR
settings.h, 40
LCD_BITMAP, 7
bpc, 7
data, 8
height, 8
size, 8
width, 8
LCD_GRAPHICS, 8
dirty, 9
dirty_x0, 9
dirty_x1, 9
dirty_y0, 9
dirty_y1, 10
graphicsmode, 10
LCD_LOGGING
settings.h, 40
LCD_MEM_READ
settings.h, 40
LCD_MEMORY
charset.c, 17
charset.h, 18
settings.h, 40
lcd.c, 26
_lcd_a0_set, 31
_lcd_char, 27
_lcd_charwidth, 27
_lcd_data, 27
_lcd_height, 31
_lcd_linewidth, 27
_lcd_reset_set, 31
_lcd_select_set, 31
_lcd_spi, 31
_lcd_width, 31
lcd_clear, 28
lcd_command, 28
lcd_init, 28
lcd_printbitmap, 29
lcd_printgraphicbitmap, 29
lcd_printtext, 29
lcd_setlogfunction, 30
lcd_textsize, 30
lcd.h, 31
lcd_clear, 33
lcd_command, 33
lcd_commands, 32
lcd_init, 33
lcd_printbitmap, 34
lcd_printgraphicbitmap, 34
lcd_printtext, 36
lcd_setlogfunction, 36
lcd_textsize, 36
lcd_bitmapalloc
bitmap.c, 11
bitmap.h, 14
lcd_bitmapclear
bitmap.c, 12
bitmap.h, 14
lcd_bitmapfree
bitmap.c, 12
bitmap.h, 15
lcd_clear
lcd.c, 28
lcd.h, 33
lcd_command
lcd.c, 28
lcd.h, 33
lcd_commands
lcd.h, 32
lcd_delay_callback
callbacks.h, 16
lcd_drawcircle
graphics.c, 20
graphics.h, 24
lcd_drawline
graphics.c, 21
graphics.h, 24
lcd_drawpoint
graphics.c, 21
graphics.h, 25
lcd_drawtext
graphics.c, 22
graphics.h, 25
lcd_fillrect
graphics.c, 22
graphics.h, 26
lcd_getbitmapmemusage
bitmap.c, 12
bitmap.h, 15
lcd_graphicsmode
graphics.h, 23
lcd_init
lcd.c, 28
lcd.h, 33
lcd_log_callback
callbacks.h, 16
lcd_logchar
log.c, 37
log.h, 38
lcd_logint
log.c, 37
log.h, 39
lcd_logstr
log.c, 37
log.h, 39
lcd_pin_callback
callbacks.h, 16

lcd_printbitmap
    lcd.c, 29
    lcd.h, 34
lcd_printgraphicbitmap
    lcd.c, 29
    lcd.h, 34
lcd_printtext
    lcd.c, 29
    lcd.h, 36
lcd_ramcheck
    log.c, 38
    log.h, 39
lcd_setlogfunction
    lcd.c, 30
    lcd.h, 36
lcd_settotalbitmapmemlimit
    bitmap.c, 13
    bitmap.h, 15
lcd_spi_callback
    callbacks.h, 17
lcd_textsize
    lcd.c, 30
    lcd.h, 36
log.c, 37
    _lcd_log, 38
    lcd_logchar, 37
    lcd_logint, 37
    lcd_logstr, 37
    lcd_ramcheck, 38
log.h, 38
    _lcd_log, 39
    lcd_logchar, 38
    lcd_logint, 39
    lcd_logstr, 39
    lcd_ramcheck, 39

MAX
    graphics.c, 19
MIN
    graphics.c, 19

README.md, 40
Release/bitmap.d, 40
Release/charset.d, 40
Release/graphics.d, 40
Release/lcd.d, 40
Release/log.d, 40

settings.h, 40
    LCD_ARCH_AVR, 40
    LCD_LOGGING, 40
    LCD_MEM_READ, 40
    LCD_MEMORY, 40
size
    LCD_BITMAP, 8

width
    LCD_BITMAP, 8