LCD_protocol100 Reference Manual

Generated by Doxygen 1.8.5

Tue Dec 10 2013 16:38:41

Contents

Chapter 1

LCD_screen Library Suite Documentation

LCD_screen is a modular suite of libraries for screens.

The LCD_screen Library Suite

- supports
  - ST7735-based RobG's universal colour LCD BoosterPack Systems
  - HY28A screen
  - ILI9225B-based RobG's 2.2" LCD+Touch Panel BoosterPack
  - 4D Systems Picaso-based screens on SGC mode (∗),
  - 4D Systems Picaso-based screens on SPE mode,
- manages display and touch, SD write and read (∗),
- is based on 3 levels
  1. top-level with dedicated GUI and Graphics libraries
  2. intermediate-level with screen-specific code
  3. low-level with virtual classes
- has been tested on Arduino 1.0.x and Energia 010

(∗) roadmap, future possible enhancements

If you enjoy this library, please help me!

See how to contribute at http://embeddedcomputing.weebly.com/contact

Developed with embedXcode

Author

Rei VILO
http://embeddedcomputing.weebly.com

Date

May 20, 2013

Version

release 102

Copyright

Chapter 2

Additional documentation

This section includes additional documentation on copyright and licence, structure, initialisation, coordinates, colour, SD-card and resources

2.1 Copyright and Licence

Copyright and Licence

The LCD_screen Library Suite is shared under dual license:

- For hobbyists and for personal usage: Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)
- For professionals or organisations or for commercial usage: All rights reserved

  For any enquiry about copyright and licence, please use the contact form.

2.2 Library Suite Structure

The LCD_screen Library Suite contains three levels of libraries:

- top level end-user libraries like label, button, dialog, menu or slider with GUI.h, or graphics with Graphics.h
- intermediate level screen-specific libraries, i.e. HY28A_sccren.h
- low level virtual classes

2.3 Version management

This section details the management and control of each library part of the Serial_LCD Library Suite.

Each library has a release number that can be check at pre-processing.

Each library has its own release number.

```
*       #define LCD_FONT_RELEASE        105
*
```

The release number is checked at pre-processing

```
*       // Other libraries
*       #include "LCD_screen_font.h"
*
*       // Test
*       #if LCD_FONT_RELEASE    < 106
*       #error required LCD_FONT_RELEASE        106
*       #endif
*
```

In this example, if the LCD_screen_font library release is 105, the pre-processor prompts an error message:

```
*       #error required LCD_FONT_RELEASE        106
*
```

2.4 Screens

This section explains the different screen features used by the library.

Each screen is driven by a controller, and each controller has different features.

A readable screen allows to get the colour of one specific pixel.

```
*      uint16_t colour;
*      if (myScreen.isReadable()) {
*          colour = myScreen.readPixel(10, 10);
*      }
*
```

If the screen isn't readable, LCD_screen::isReadable() is false and LCD_screen::readPixel() returns 0.

The LCD_screen::readPixel() function is required by the LCD_screen::copyPaste() and LCD_screen::copyArea() functions.

2.5 Storage

This section explains the different kinds of storage used used by the library.

The GUI library saves the initial screen before displaying a dialog box, a menu or a slider, to restore it afterwards.

A storage can be:

- external SRAM
- SD-card

The function LCD_screen::isStorage() returns true is a storage is available.

The storage is required by the LCD_screen::copyArea() and LCD_screen::pasteArea() functions.

- LCD_screen::copyArea() copies an area from the screen and saves it to the SRAM or SD-card
- LCD_screen::pasteArea() reads an area from the SRAM or SD-card and pastes it to the screen

The MCU SRAM is used for the LCD_screen::copyPaste() function.

2.6 Coordinates systems

This section explains the rectangle and vector coordinates systems.

Two systems of coordinates are used, rectangle and vector coordinates.

Rectangle coordinates include two points P1 and P2.

- P1 is a pixel on the top left, with (x1, y1) coordinates.
- P2 is a pixel on the bottom right, with (x2, y2) coordinates.

Example rectangle (0, 0) - (319, 239)

Vector coordinates include one point P0 and one distance.

- P0 is a pixel and the origin, with (x0, y0) coordinates.
- The distance (dx, dy) is specified for the horizontal and the vertical axis.

Example vector (0, 0) - (320, 240)

Going from pixel 0 to pixel 319 represents 320 pixels in total

2.7 Fonts

This section explains how to use the fonts.

Four extended fonts are supplied:

- Font 0 or Terminal6x8e fixed 6 x 8, size= 1344 bytes, cumulated= 1344 bytes
- Font 1 or Terminal8x12e fixed 8 x 12, size= 3584 bytes, cumulated= 4928 bytes
- Font 2 or Terminal12x16e fixed 12 x 16, size= 5376 bytes, cumulated= 10304 bytes
- Font 3 or Terminal16x24e fixed 16 x 24, size= 10752 bytes, cumulated= 21056 bytes (not released)

All the fonts include the extended characters 0x80∼0xff corresponding to the ISO-8859-1 fonts page.

To convert UTF-8 strings to ISO-8859-1 strings, use the utf2iso() utility.

Note

First font is numbered 0, second 1, ... The latest font is numbered LCD_screen::fontMax()-1.
MAX_FONT_SIZE=0 means no font.

Number of fonts

Returns

number of fonts available

Note

First font is numbered 0, second 1, ...
The latest font is numbered LCD_screen_font::fontMax()-1

See Also

- MikroElektronika GLCD Font Creator 1.2.0.0
  http://www.mikroe.com
- The Unicode Consortium. The Unicode Standard, Version 6.2.0,
  (Mountain View, CA: The Unicode Consortium, 2012. ISBN 978-1-936213-07-8)
  http://www.unicode.org/versions/Unicode6.2.0/

2.8 Values using integer numbers

This section explains how values are coded using integer numbers only.

Using integers only allows to avoid loading the library for real numbers, which requires 6 KB of memory.

A value are coded using two numbers:

- a significand, int32_t number, already multiplied by unit
- plus a multiplier, int32_t unit, with default=1, 10 or 100

  value = number / unit = significand / multiplier The unit provides the scale of the degrees passed.

  The following calls of the draw() function are equivalent:

```
*      draw(90);         // = 90 / 1
*      draw(90, 1);      // = 90 / 1
*      draw(9000, 100); // = 9000 / 100
*
```

Functions like cos32x100 and sin32x100 receive and return values multiplied by 100. The unit is set at 100.

- int32_t cos32x100(int32_t degreesX100)
- int32_t sin32x100(int32_t degreesX100)

int32_t are used instead of int64_t because some platforms don't manage 64-bit numbers.

See Also

- Wikipedia on Floating points and Significand
  https://en.wikipedia.org/wiki/Floating_point and https://en.wikipedia.-org/wiki/Significand

2.9 Colours

This section explains how the colours are coded in 16-bit colours and 8-bits Red-Green-Blue components.

Colours are coded internally on 16 bits, with 5 bits for red, 6 bits for green and 5 bits for blue, or called RGB565.

The Red-Green-Blue components are 8-bit sized and 0x00..0xff scaled.

Two functions are available to convert 16-bit colours and 8-bit Red-Green-Blue components:

- LCD_screen::calculateColour calculates 16-bit colour from 8-bit Red-Green-Blue components
- LCD_screen::splitColour calculates 8-bit Red-Green-Blue components from 16-bit colour

  Two functions provide additional calculations:

- LCD_screen::halveColour halves a 16-bit colour
- LCD_screen::reverseColour reverses a 16-bit colour

See Also

from Embedded Computing website:

- Intermediate Level: Colour Functions

2.10 Other resources

More resources are available online.

The LCD_screen Library Suite is supported by the dedicated Embedded Computing website at http://embeddedcomputing.weebly.com

See Also

from Embedded Computing website:

- Main page
- Download
- Former Tutorials
- Former Examples
- Former Tutorial 3: FAQ
- Former LCD_screen Library Suite
- Fonts and font generator except for Terminal16e font

Chapter 3

Version history

Version history for LCD_screen

3.1 LCD_screen.h

- May 26, 2013 release 104 Built-in fonts and separate LCD_screen_font.h
- May 26, 2013 release 105 Virtual functions
- May 26, 2013 release 106 Initial release
- Jul 02, 2013 release 107 SRAM integration
- Jul 06, 2013 release 108 SRAM speed optimisation
- Jul 10, 2013 release 109 GUI integration
- Aug 16, 2013 release 110 Storage integration
- Oct 26, 2013 release 113 New screen added
- Dec 10, 2013 release 114 Text functions refactoring

3.2 LCD_screen_font.h

Version history for LCD_screen_font

- May 26, 2013 release 103 Virtual functions
- May 26, 2013 release 104 Integration of LCD_font
- May 26, 2013 release 105 Initial release
- May 26, 2013 release 106 Initial release
- Jul 02, 2013 release 107 SRAM integration
- Jul 06, 2013 release 108 SRAM speed optimisation
- Jul 10, 2013 release 109 GUI integration
- Aug 16, 2013 release 110 Storage integration
- Aug 24, 2013 release 111 Stability enhancement
- Sep 08, 2013 release 112 uint8_t for unsigned char

- Oct 26, 2013 release 113 New screen added
- Dec 10, 2013 release 114 Text functions refactoring

3.3 LCD_utilities.h

Version history for the utilities

- May 10, 2012 release 100 Initial release
- Jul 10, 2013 release 101 Better algorithms
- Sep 18, 2013 release 102 Use of char[] and C functions

3.4 Screen_K35.h

Version history for the Kentec 3.5 screen

- Aug 16, 2013 release 103 initial release

3.5 Screen_HX8353.h

Version history for the ST7735-based screen

- May 26, 2013 release 105 initial release

3.6 Screen_HX8353.h

Version history for the HX8353-based screen

- Dec 06, 2013 release 101 initial release

3.7 Screen_ILI9225B.h

Version history for the ILI9225B-based screen

- May 26, 2013 release 105 SPI speed fixed for screen and touch
- May 26, 2013 release 105 Faster text (10x)
- May 26, 2013 release 106 Initial release
- May 26, 2013 release 107 gText fixed
- Sep 09, 2013 release 108 Added support for F5529

3.8 Screen_HY28A_SRAM.h

Version history for the HY28A-based screen

- May 26, 2013 release 105 Dual SPI
- May 26, 2013 release 106 Faster text (6x)
- May 26, 2013 release 107 Initial release
- May 26, 2013 release 108 _setPoint fixed
- Jun 02, 2013 release 109 Fast software SPI evaluation
- Jul 02, 2013 release 110 SRAM copy paste
- Jul 07, 2013 release 111 Improved SPI library by reaper7
- Aug 10, 2013 release 112 Improved SRAM management

3.9 Screen_HI32.h

Version history for the HY28A-based screen

- May 29, 2013 release 101 stable release

3.10 Screen_W32.h

Version history for the 3.2" wide screen

- May 29, 2013 release 101 Stable release
- Oct 05, 2013 release 102 Fix for orientation

3.11 Screen_HX8353E.h

Version history for the 3.2" wide screen

- Dec 10, 2013 release 100 Educational BoosterPack MKII (not released)

3.12 Screen_PicasoSGC.h

Version history for 4D Systems Picaso-based screen on SGC mode

- May 27, 2013 release 099 Proof of concept (not released)
- Jun 25, 2013 release 100 Interim version (not released)
- Jun 25, 2013 release 101 Interim version (not released)
- Jun 25, 2013 release 102 Interim version (not released)
- Sep 25, 2013 release 103 readPixel and copyPaste added (not released)

3.13 Screen_PicasoSPE.h

Version history for 4D Systems Picaso-based screen on SPE mode

- May 27, 2013 release 099 Proof of concept (not released)
- Jun 09, 2013 release 100 Proof of concept (not released)
- Jun 09, 2013 release 101 Proof of concept (not released)
- Jun 09, 2013 release 102 Proof of concept (not released)
- Jun 09, 2013 release 103 Proof of concept (not released)
- Jun 09, 2013 release 104 Proof of concept (not released)
- Sep 25, 2013 release 105 readPixel and copyPaste added (not released)
- Oct 20, 2013 release 106 Support for Energia
- Oct 20, 2013 release 106 First release

3.14 LCD_graphics.h

Version history for the HY28A-based screen

- Jan 22, 2012 release 1 New Graphics library with example Graphics_main
- Jan 25, 2012 release 2 (x0, y0, dx, fy) functions added to (x1, x2, y1, y2) = (x0, y0, x0+dx, y0+dy)
- Jan 27, 2012 release 3 Histogram graphic with example Histogram_main
- Jan 28, 2012 release 103 New index
- Jan 28, 2012 release 104 Default colours for each graphic
- Jan 30, 2012 release 105 Improved consistency
- Feb 01, 2012 release 106 ttoa used
- Feb 12, 2012 release 106a **AVR_ATmega328P** added
- Feb 14, 2012 release 107 gGauge graphic
- Feb 16, 2012 release 108 Yaw, pitch, roll graphics
- Mar 19, 2012 release 209 Arduino 1.0 compatible
- May 01, 2012 release 109 Support for Wiring 1.0
- Jun 14, 2012 release 110 Unified library for Arduino 23 & 1.0, chipKIT and Wiring
- Jul 06, 2012 release 111 More compact library
- Jul 08, 2012 release 112 Meta-classes mtgDial and mtfPane
- Jul 09, 2012 release 113 Graphic for multiple values
- Jul 13, 2012 release 114 Better dial for clock
- Jul 30, 2012 release 314 Unified release numbering version 3xx
- Aug 21, 2012 release 315 **AVR_ATmega1280** added
- Sep 01, 2012 release 316 **AVR_ATmega32U4** added

- Sep 12, 2012 release 317 Gauge stability enhancement
- Mar 03, 2013 release 318 gTable graphic
- Jun 03, 2013 release 419 clock, gauge and yaw-pitch-roll for LCD_screen
- Jun 04, 2013 release 420 Full library for LCD_screen

3.15 LCD_GUI.h

Version history for the HY28A-based screen

- Sep 18, 2011 release 1 Dialog window with up to 3 buttons
- Nov 25, 2011 release 2 Faster dialog show/hide and optional area for screen copy to/read from SD
- Nov 27, 2011 release 3 Bugs fixed
- Dec 15, 2011 release 3.1 Arduino 1.0 implementation test no longer compatible with 0022
- Dec 27, 2011 release 4 Ready for GUI = button + dialog box + menu + label
- Dec 28, 2011 release 5 Item-based refactoring for dialog box, menu and label
- Dec 29, 2011 release 6 Button library deprecated, superseeded by GUI library
- Jan 05, 2012 release 7 (x0, y0, dx, fy) functions added to (x1, x2, y1, y2) = (x0, y0, x0+dx, y0+dy)
- Jan 25, 2012 release 8 Button with instant option (no de-bouncing)
- Jan 28, 2012 release 108 New index
- Feb 12, 2012 release 108a **AVR_ATmega328P AVR_ATmega2560** added
- Mar 19, 2012 release 209 Arduino 1.0 compatible
- Apr 22, 2012 release 109 Slider added
- Apr 28, 2012 release 110 Better menu
- May 01, 2012 release 111 Support for Wiring 1.0
- Jun 04, 2012 release 112 RAW image bug fixed
- Jun 14, 2012 release 113 Unified library for Arduino 23 & 1.0, chipKIT and Wiring
- Jul 05, 2012 release 114 More compact library
- Jul 30, 2012 release 315 Unified release numbering version 3xx
- Aug 21, 2012 release 316 **AVR_ATmega1280** added
- Sep 01, 2012 release 317 **AVR_ATmega32U4** added
- Dec 01, 2012 release 318 New area object = zone for touch
- Jan 08, 2013 release 319 New cursor object
- Jan 15, 2013 release 320 New text box object
- Jul 06, 2013 release 421 First release for LCD_screen

3.16 Fonts

Version history for fonts

- May 25, 2013 release 101 Initial sets of fonts Terminal6, Terminal8 and Terminal12
- May 27, 2013 release 102 Extended sets of fonts Terminal6e, Terminal8e and Terminal12e
- Jun 04, 2013 release 103 Added 16x24 font Terminal16e (not released)

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 5

Class Index

5.1 Class List

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

Chapter 6

File Index

6.1 File List

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

Chapter 7

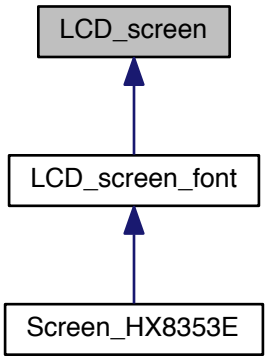
Class Documentation

7.1 LCD_screen Class Reference

Generic LCD class.

`#include <LCD_screen.h>`

Inheritance diagram for LCD_screen:



Public Member Functions

- LCD_screen ()
  Constructor.

General

- virtual void begin ()=0
  Initialisation.
- virtual String WhoAmI ()=0
  Request information about the screen.
- void clear (uint16_t colour=blackColour)
  Clear the screen.
- virtual void setOrientation (uint8_t orientation)

Set orientation.

- uint8_t getOrientation ()

  Get orientation.
- virtual void showInformation (uint16_t x0=0, uint16_t y0=0)

  Show information.
- virtual uint16_t screenSizeX ()

  Screen size, x-axis.
- virtual uint16_t screenSizeY ()

  Screen size, y-axis.

Graphics

- virtual void circle (uint16_t x0, uint16_t y0, uint16_t radius, uint16_t colour)

  Draw circle.
- virtual void arc (uint16_t x0, uint16_t y0, uint16_t radius, uint16_t start, uint16_t end, uint16_t colour)

  Draw arc.
- virtual void line (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2, uint16_t colour)

  Draw line, rectangle coordinates.
- virtual void dLine (uint16_t x0, uint16_t y0, uint16_t dx, uint16_t dy, uint16_t colour)

  Draw line, vector coordinates.
- virtual void setPenSolid (bool flag=true)

  Set pen opaque.
- virtual void triangle (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2, uint16_t x3, uint16_t y3, uint16_t colour)

  Draw triangle, rectangle coordinates.
- virtual void rectangle (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2, uint16_t colour)

  Draw rectangle, rectangle coordinates.
- virtual void dRectangle (uint16_t x0, uint16_t y0, uint16_t dx, uint16_t dy, uint16_t colour)

  Draw rectangle, vector coordinates.
- virtual void point (uint16_t x1, uint16_t y1, uint16_t colour)

  Draw pixel.

Text

Read pixel colour

Parameters

x1	*point coordinate, x-axis*
y1	*point coordinate, y-axis*

Returns

16-bit colour, bits 15-11 red, bits 10-5 green, bits 4-0 blue

- virtual void setFontSize (uint8_t size)=0

  Select font size.
- virtual void setFontSolid (bool flag=true)

  Set transparent or opaque text.
- virtual uint8_t fontSizeX ()=0

  Font size, x-axis.
- virtual uint8_t fontSizeY ()=0

  Font size, y-axis.
- virtual void gText (uint16_t x0, uint16_t y0, String s, uint16_t textColour=whiteColour, uint16_t backColour=blackColour, uint8_t ix=1, uint8_t iy=1)=0

  Draw ASCII Text (pixel coordinates) with selection of size.

Colours utilities

- uint16_t calculateColour (uint8_t red, uint8_t green, uint8_t blue)

  Calculate 16-bit colour from 8-bit Red-Green-Blue components.

- void splitColour (uint16_t rgb, uint8_t &red, uint8_t &green, uint8_t &blue)
  Calculate 8-bit Red-Green-Blue components from 16-bit colour.
- uint16_t halveColour (uint16_t rgb)
  Half 16-bit colour.
- uint16_t averageColour (uint16_t rgb1, uint16_t rgb2)
  Average two 16-bit colours.
- uint16_t reverseColour (uint16_t rgb)
  Reverse 16-bit colour.

Advanced features

- bool isReadable ()
  Is screen readable?
- bool isStorage ()
  Does the screen feature an external storage?
- virtual uint16_t readPixel (uint16_t x1, uint16_t y1)
  Read pixel colour.
- virtual void copyPaste (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2, uint16_t dx, uint16_t dy)
  Copy a source area to a target area.
- virtual void copyArea (uint16_t x0, uint16_t y0, uint16_t dx, uint16_t dy, uint32_t &address)
  Copy an area to an external support.
- virtual void pasteArea (uint16_t x0, uint16_t y0, uint16_t dx, uint16_t dy, uint32_t &address, bool option=false)
  Paste an area from an external support.

Touch

- bool isTouch ()
  Is touch available?
- bool getTouch (uint16_t &x, uint16_t &y, uint16_t &z)
  Poll touch.
- void calibrateTouch ()
  Calibrate the touch.

Protected Member Functions

- virtual void **_fastFill** (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2, uint16_t colour)=0
- virtual void **_setPoint** (uint16_t x1, uint16_t y1, uint16_t colour)=0
- virtual void **_getRawTouch** (uint16_t &x0, uint16_t &y0, uint16_t &z0)=0
- virtual void **_setWindow** (uint16_t x0, uint16_t y0, uint16_t x1, uint16_t y1)=0
- virtual void **_writeData88** (uint8_t dataHigh8, uint8_t dataLow8)=0
- void **_displayTarget** (uint16_t x0, uint16_t y0, uint16_t colour)
- void **_swap** (int16_t &a, int16_t &b)
- void **_swap** (uint16_t &a, uint16_t &b)
- void **_swap** (uint8_t &a, uint8_t &b)
- uint16_t **_check** (uint16_t x0, uint16_t xmin, uint16_t xmax)
- void **_triangleArea** (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2, uint16_t x3, uint16_t y3, uint16_t colour)
- bool **_inValue** (int16_t value, int16_t valueLow, int16_t valueHigh)
- bool **_inSector** (int16_t valueStart, int16_t valueEnd, int16_t sectorLow, int16_t sectorHigh, int16_t criteriaStart, int16_t criteriaEnd, int16_t criteriaLow, int16_t criteriaHigh, int16_t criteria)
- bool **_inCycle** (int16_t value, int16_t valueLow, int16_t valueHigh)

Protected Attributes

- uint8_t **_fontX**
- uint8_t **_fontY**
- uint8_t **_fontSize**
- uint8_t **_orientation**
- bool **_penSolid**
- bool **_fontSolid**
- bool **_flagRead**
- bool **_flagStorage**
- uint16_t **_screenWidth**
- uint16_t **_screenHeigth**
- uint8_t **_touchTrim**
- uint16_t **_touchXmin**
- uint16_t **_touchXmax**
- uint16_t **_touchYmin**
- uint16_t **_touchYmax**

7.1.1 Detailed Description

Generic LCD class.

7.1.2 Member Function Documentation

7.1.2.1 void LCD_screen::arc (uint16_t *x0,* uint16_t *y0,* uint16_t *radius,* uint16_t *start,* uint16_t *end,* uint16_t *colour*) [virtual]

Draw arc.

Parameters

x0	center, point coordinate, x-axis
y0	center, point coordinate, y-axis
radius	radius
start	starting angle, in degrees
end	ending angle, in degrees
colour	16-bit colour

Note

if ending angle $<$ starting angle, then starting angle..360 and 0..starting angle arcs are drawn

7.1.2.2 uint16_t LCD_screen::averageColour (uint16_t *rgb1,* uint16_t *rgb2*)

Average two 16-bit colours.

Parameters

rgb1	first 16-bit colour
rgb2	second 16-bit colour

Returns

averaged 16-bit colour

More: Colours

7.1.2.3 virtual void LCD_screen::begin ( ) [pure virtual]

Initialisation.

Warning

Definition for this method is compulsory.

Implemented in Screen_HX8353E.

7.1.2.4 uint16_t LCD_screen::calculateColour (uint8_t *red*, uint8_t *green*, uint8_t *blue*)

Calculate 16-bit colour from 8-bit Red-Green-Blue components.

Parameters

red	red component, 0x00..0xff
green	green component, 0x00..0xff
blue	blue component, 0x00..0xff

Returns

16-bit colour

More: Colours

7.1.2.5 void LCD_screen::circle (uint16_t *x0*, uint16_t *y0*, uint16_t *radius*, uint16_t *colour*) [virtual]

Draw circle.

Parameters

x0	center, point coordinate, x-axis
y0	center, point coordinate, y-axis
radius	radius
colour	16-bit colour

7.1.2.6 void LCD_screen::copyArea (uint16_t *x0*, uint16_t *y0*, uint16_t *dx*, uint16_t *dy*, uint32_t & *address*) [virtual]

Copy an area to an external support.

Parameters

x0	source top left coordinate, x-axis
y0	source top left coordinate, y-axis
dx	width to be copied, x-axis
dy	eight to be copied, y-axis
address	identifier, as SRAM address or file number

Note

This feature requires a readable screen and a storage.

More: Coordinates systems, Screens, Storage

7.1.2.7 void LCD_screen::copyPaste (uint16_t *x1*, uint16_t *y1*, uint16_t *x2*, uint16_t *y2*, uint16_t *dx*, uint16_t *dy*) [virtual]

Copy a source area to a target area.

Parameters

x1	source top left coordinate, x-axis
y1	source top left coordinate, y-axis
x2	target top left coordinate, x-axis
y2	target top left coordinate, y-axis
dx	width to be copied, x-axis
dy	eight to be copied, y-axis

Note

This feature requires a readable screen.

Warning

The function doesn't manage the overlapping of the source and target areas. If such a case, use copyArea() pasteArea() instead.

More: Coordinates systems, Screens

7.1.2.8 void LCD_screen::dLine (uint16_t *x0,* uint16_t *y0,* uint16_t *dx,* uint16_t *dy,* uint16_t *colour*) [virtual]

Draw line, vector coordinates.

Parameters

x0	point coordinate, x-axis
y0	point coordinate, y-axis
dx	length, x-axis
dy	height, y-axis
colour	16-bit colour

More: Coordinates systems, Colours

7.1.2.9 void LCD_screen::dRectangle (uint16_t *x0,* uint16_t *y0,* uint16_t *dx,* uint16_t *dy,* uint16_t *colour*) [virtual]

Draw rectangle, vector coordinates.

Parameters

x0	point coordinate, x-axis
y0	point coordinate, y-axis
dx	length, x-axis
dy	height, y-axis
colour	16-bit colour

More: Coordinates systems, Colours

7.1.2.10 virtual uint8_t LCD_screen::fontSizeX ( ) [pure virtual]

Font size, x-axis.

Returns

horizontal size of current font, in pixels

Warning

Definition for this method is compulsory.

Implemented in LCD_screen_font.

7.1.2.11 virtual uint8_t LCD_screen::fontSizeY ( ) [pure virtual]

Font size, y-axis.

Returns

vertical size of current font, in pixels

Warning

Definition for this method is compulsory.

Implemented in LCD_screen_font.

7.1.2.12 uint8_t LCD_screen::getOrientation ( )

Get orientation.

Returns

orientation orientation, 0 = portrait, 1 = right rotated landscape, 2 = reverse portrait, 3 = left rotated landscape

7.1.2.13 bool LCD_screen::getTouch (uint16_t & *x*, uint16_t & *y*, uint16_t & *z*)

Poll touch.

Parameters

x	x coordinate
y	y coordinate
z	z coordinate=pressure

Returns

true if pressed

7.1.2.14 virtual void LCD_screen::gText (uint16_t *x0*, uint16_t *y0*, String *s*, uint16_t *textColour* = **whiteColour**, uint16_t *backColour* = **blackColour**, uint8_t *ix* = 1, uint8_t *iy* = 1) [pure virtual]

Draw ASCII Text (pixel coordinates) with selection of size.

Parameters

x0	point coordinate, x-axis
y0	point coordinate, y-axis

s	text string
textColour	16-bit colour, default = white
backColour	16-bit colour, default = black
ix	x-axis font size multiplier, default = 1
iy	y-axis font size multiplier, default = 1

Warning

Definition for this method is compulsory.
More: Colours

Implemented in LCD_screen_font.

7.1.2.15 uint16_t LCD_screen::halveColour (uint16_t *rgb*)

Half 16-bit colour.

Parameters

rgb	16-bit colour

Returns

halved 16-bit colour

More: Colours

7.1.2.16 bool LCD_screen::isReadable ( )

Is screen readable?

Returns

true is screen readable, false otherwise

More: Screens

7.1.2.17 bool LCD_screen::isStorage ( )

Does the screen feature an external storage?

Returns

true is storage available, false otherwise

More: Storage

7.1.2.18 bool LCD_screen::isTouch ( )

Is touch available?

Returns

true is touch available, false otherwise

7.1.2.19 void LCD_screen::line (uint16_t *x1*, uint16_t *y1*, uint16_t *x2*, uint16_t *y2*, uint16_t *colour*) [virtual]

Draw line, rectangle coordinates.

Parameters

x1	top left coordinate, x-axis
y1	top left coordinate, y-axis
x2	bottom right coordinate, x-axis
y2	bottom right coordinate, y-axis
colour	16-bit colour

7.1.2.20 void LCD_screen::pasteArea (uint16_t *x0,* uint16_t *y0,* uint16_t *dx,* uint16_t *dy,* uint32_t & *address,* bool *option* = `false`) [virtual]

Paste an area from an external support.

Parameters

x0	source top left coordinate, x-axis
y0	source top left coordinate, y-axis
dx	target to be pasted, x-axis
dy	target to be pasted, y-axis
address	identifier, as SRAM address or file number
option	false=default=original colours, true=halved colours

Note

This feature requires a readable screen and a storage.

More: Coordinates systems, Screens, Storage

7.1.2.21 void LCD_screen::point (uint16_t *x1,* uint16_t *y1,* uint16_t *colour*) [virtual]

Draw pixel.

Parameters

x1	point coordinate, x-axis
y1	point coordinate, y-axis
colour	16-bit colour

More: Coordinates systems, Colours

7.1.2.22 uint16_t LCD_screen::readPixel (uint16_t *x1,* uint16_t *y1*) [virtual]

Read pixel colour.

Parameters

x1	point coordinate, x-axis
y1	point coordinate, y-axis

Returns

16-bit colour, bits 15-11 red, bits 10-5 green, bits 4-0 blue

Note

This feature requires a readable screen.

More: Coordinates systems, Colours, Screens

7.1.2.23 void LCD_screen::rectangle (uint16_t *x1*, uint16_t *y1*, uint16_t *x2*, uint16_t *y2*, uint16_t *colour*) [virtual]

Draw rectangle, rectangle coordinates.

Parameters

x1	top left coordinate, x-axis
y1	top left coordinate, y-axis
x2	bottom right coordinate, x-axis
y2	bottom right coordinate, y-axis
colour	16-bit colour

More: Coordinates systems, Colours

7.1.2.24 uint16_t LCD_screen::reverseColour (uint16_t *rgb*)

Reverse 16-bit colour.

Parameters

rgb	16-bit colour

Returns

reversed 16-bit colour

More: Colours

7.1.2.25 uint16_t LCD_screen::screenSizeX ( ) [virtual]

Screen size, x-axis.

Returns

horizontal size of the screen, in pixels

Note

240 means 240 pixels and thus 0..239 coordinates (decimal)

7.1.2.26 uint16_t LCD_screen::screenSizeY ( ) [virtual]

Screen size, y-axis.

Returns

vertical size of the screen, in pixels

Note

240 means 240 pixels and thus 0..239 coordinates (decimal)

7.1.2.27 virtual void LCD_screen::setFontSize (uint8_t *size*) [pure virtual]

Select font size.

Parameters

size	default = 0 = small, 1 = large

Warning

Definition for this method is compulsory.

Implemented in LCD_screen_font.

7.1.2.28 void LCD_screen::setFontSolid (bool *flag* = `true`) [virtual]

Set transparent or opaque text.

Parameters

flag	default = 1 = opaque = solid, false = transparent

Warning

Definition for this method is compulsory.

7.1.2.29 void LCD_screen::setOrientation (uint8_t *orientation*) [virtual]

Set orientation.

Parameters

orientation	orientation, 0 = portrait, 1 = right rotated landscape, 2 = reverse portrait, 3 = left rotated landscape

Reimplemented in Screen_HX8353E.

7.1.2.30 void LCD_screen::setPenSolid (bool *flag* = `true`) [virtual]

Set pen opaque.

Parameters

flag	default = true = opaque = solid, false = wire frame

7.1.2.31 void LCD_screen::showInformation (uint16_t *x0* = `0`, uint16_t *y0* = `0`) [virtual]

Show information.

Parameters

x0	left coordinate, x-axis, default=0
y0	top coordinate, y-axis, default=0

Display information: screen, size, fonts, touch

7.1.2.32 void LCD_screen::splitColour (uint16_t *rgb*, uint8_t & *red*, uint8_t & *green*, uint8_t & *blue*)

Calculate 8-bit Red-Green-Blue components from 16-bit colour.

Parameters

rgb	16-bit colour
red	red component, 0x00..0xff
green	green component, 0x00..0xff
blue	blue component, 0x00..0xff

More: Colours

7.1.2.33 void LCD_screen::triangle (uint16_t *x1*, uint16_t *y1*, uint16_t *x2*, uint16_t *y2*, uint16_t *x3*, uint16_t *y3*, uint16_t *colour*) [virtual]

Draw triangle, rectangle coordinates.

Parameters

x1	first point coordinate, x-axis
y1	first point coordinate, y-axis
x2	second point coordinate, x-axis
y2	second point coordinate, y-axis
x3	third point coordinate, x-axis
y3	third point coordinate, y-axis
colour	16-bit colour **More:** Coordinates systems, Colours

7.1.2.34 virtual String LCD_screen::WhoAmI ( ) [pure virtual]

Request information about the screen.

Returns

string with hardware version

Warning

Definition for this method is compulsory.

Implemented in Screen_HX8353E.

The documentation for this class was generated from the following files:

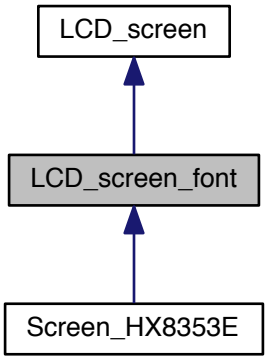
- LCD_screen.h
- LCD_screen.cpp

7.2 LCD_screen_font Class Reference

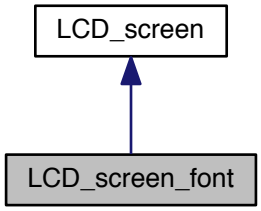
Generic LCD with font class.

`#include <LCD_screen_font.h>`

Inheritance diagram for LCD_screen_font:



Collaboration diagram for LCD_screen_font:



Public Member Functions

- LCD_screen_font ()
  Constructor.

Text

- virtual void setFontSize (uint8_t font=0)
  Set font size.
- virtual uint8_t fontMax ()
  Number of fonts.
- virtual uint8_t fontSizeX ()
  Font size, x-axis.
- virtual uint8_t fontSizeY ()
  Font size, y-axis.
- virtual void gText (uint16_t x0, uint16_t y0, String s, uint16_t textColour=whiteColour, uint16_t backColour=blackColour, uint8_t ix=1, uint8_t iy=1)
  Draw ASCII Text (pixel coordinates) with selection of size.

Protected Member Functions

- uint8_t **_getCharacter** (uint8_t c, uint8_t i)
- virtual void **_fastFill** (uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2, uint16_t colour)=0
- virtual void **_setPoint** (uint16_t x1, uint16_t y1, uint16_t colour)=0
- virtual void **_getRawTouch** (uint16_t &x0, uint16_t &y0, uint16_t &z0)=0
- virtual void **_setWindow** (uint16_t x0, uint16_t y0, uint16_t x1, uint16_t y1)=0
- virtual void **_writeData88** (uint8_t dataHigh8, uint8_t dataLow8)=0

Additional Inherited Members

7.2.1 Detailed Description

Generic LCD with font class.

7.2.2 Member Function Documentation

7.2.2.1 uint8_t LCD_screen_font::fontMax ( ) [virtual]

Number of fonts.

Returns

number of fonts available

Note

First font is numbered 0, second 1, ...
The latest font is numbered fontMax()-1

7.2.2.2 uint8_t LCD_screen_font::fontSizeX ( ) [virtual]

Font size, x-axis.

Returns

horizontal size of current font, in pixels

Implements LCD_screen.

7.2.2.3 uint8_t LCD_screen_font::fontSizeY ( ) [virtual]

Font size, y-axis.

Returns

vertical size of current font, in pixels

Implements LCD_screen.

7.2.2.4 void LCD_screen_font::gText (uint16_t *x0*, uint16_t *y0*, String *s*, uint16_t *textColour* = **whiteColour**, uint16_t *backColour* = **blackColour**, uint8_t *ix* = 1, uint8_t *iy* = 1) [virtual]

Draw ASCII Text (pixel coordinates) with selection of size.

Parameters

x0	point coordinate, x-axis
y0	point coordinate, y-axis
s	text string
textColour	16-bit colour, default = white
backColour	16-bit colour, default = black
ix	x-axis font size multiplier, default = 1
iy	y-axis font size multiplier, default = 1

Implements LCD_screen.

7.2.2.5 void LCD_screen_font::setFontSize (uint8_t *font* = 0) [virtual]

Set font size.

Parameters

font	default=0=small, 1=larger, up to fontMax()-1

Implements LCD_screen.

The documentation for this class was generated from the following files:

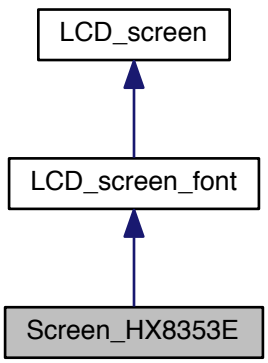
- LCD_screen_font.h
- LCD_screen_font.cpp

7.3 Screen_HX8353E Class Reference

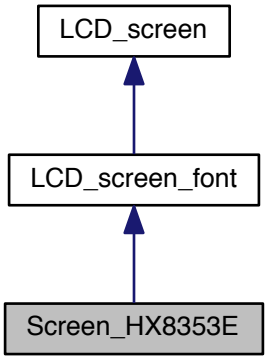
Class for Educational BoosterPack MKII.

`#include <Screen_HX8353E.h>`

Inheritance diagram for Screen_HX8353E:



Collaboration diagram for Screen_HX8353E:



Public Member Functions

- Screen_HX8353E ()

  Constructor with default pins.
- Screen_HX8353E (uint8_t resetPin, uint8_t dataCommandPin, uint8_t chipSelectPin, uint8_t backlightPin)

  Constructor.
- void begin ()

  Initialisation.
- String WhoAmI ()

  Request information about the screen.
- void invert (boolean flag)

  Invert screen.
- void setBacklight (boolean flag)

  Switch backlight on or off.
- void setDisplay (boolean flag)

  Switch display on or off.
- void setOrientation (uint8_t orientation)

  Set orientation.

Additional Inherited Members

7.3.1 Detailed Description

Class for Educational BoosterPack MKII.

Screen controller

- LCD: HX8353E, 4-wire 8-bit SPI with R/S line
- touch: no touch

7.3.2 Constructor & Destructor Documentation

7.3.2.1 Screen_HX8353E::Screen_HX8353E ( )

Constructor with default pins.

Note

Default pins for LaunchPad MSP430F5529 / LaunchPad Stellaris LM4F
17 / NULL / NULL = LCD Reset
31 / P?_? / P?_? = LCD Data/Command
13 / P?_? / P?_? = LCD Chip Select
39 / P?_? / P?_? = LCD PWM Backlight

7.3.2.2 Screen_HX8353E::Screen_HX8353E (uint8_t *resetPin,* uint8_t *dataCommandPin,* uint8_t *chipSelectPin,* uint8_t *backlightPin*)

Constructor.

Parameters

resetPin	digital pin number for screen reset
dataCommand-Pin	digital pin number for command / data
chipSelectPin	digital pin number for SPI chip select
backlightPin	PWM pin number for backlight

7.3.3 Member Function Documentation

7.3.3.1 void Screen_HX8353E::invert (boolean *flag*)

Invert screen.

Parameters

flag	true to invert, false for normal screen

7.3.3.2 void Screen_HX8353E::setBacklight (boolean *flag*)

Switch backlight on or off.

Parameters

flag	true=on, false=off

7.3.3.3 void Screen_HX8353E::setDisplay (boolean *flag*)

Switch display on or off.

Parameters

flag	true=on, false=off

7.3.3.4 void Screen_HX8353E::setOrientation (uint8_t *orientation*) [virtual]

Set orientation.

Parameters

orientation	orientation, 0=portrait, 1=right rotated landscape, 2=reverse portrait, 3=left rotated landscape

Reimplemented from LCD_screen.

7.3.3.5 String Screen_HX8353E::WhoAmI ( ) [virtual]

Request information about the screen.

Returns

string with hardware version

Implements LCD_screen.

The documentation for this class was generated from the following files:

- Screen_HX8353E.h
- Screen_HX8353E.cpp

Chapter 8

File Documentation

8.1 LCD_documentation.h File Reference

Documentation for the LCD_screen Library Suite.

8.1.1 Detailed Description

Documentation for the LCD_screen Library Suite. Additional documentation on coordinates, fonts and colours

The LCD_screen Library Suite is the continuation of the `Serial_LCD Library Suite`. The Serial_LCD Library Suite is now obsolete and no longer maintained. 4D Systems has launched a new series of screens and provides the libraries for the new serial SPE2 protocol, which is not compatible with former SGC serial protocol. While Serial_LCD Library Suite was limited to 4D Systems screens in SGC mode, the LCD_screen Library Suite adresses a larger audience of basic screens, with a variety of sizes, connections and features.

Member of LCD_screen Library Suite

LCD_screen Library Suite

For Arduino 1.0, chipKIT MPIDE 0023, Wiring 1.0, Energia 009

Developed with `embedXcode`

Author

Rei VILO
`http://embeddedcomputing.weebly.com`

Date

May 20, 2013

Version

release 109

Copyright

Dual license:

- For hobbyists and for personal usage: Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)
- For professionals or organisations or for commercial usage: All rights reserved
- For hobbyists and for personal usage: Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)

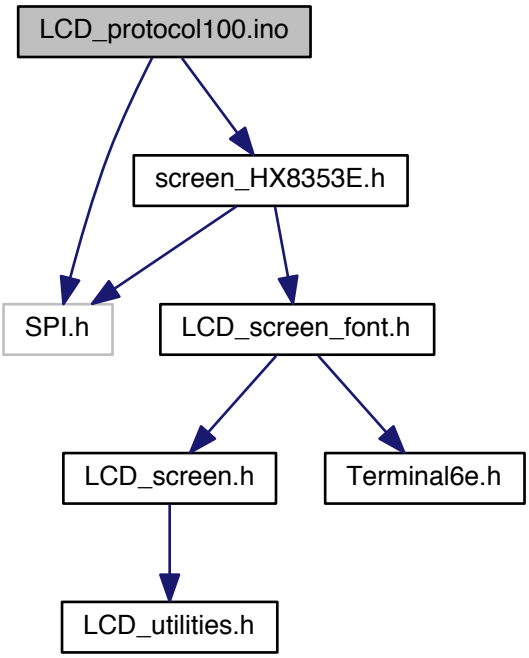
For any enquiry about license, http://embeddedcomputing.weebly.com/contact

8.2 LCD_protocol100.ino File Reference

Main sketch.

```
#include "SPI.h"
#include "screen_HX8353E.h"
```

Include dependency graph for LCD_protocol100.ino:



Functions

- void protocolSquare (uint16_t pixels)

  protocolSquare
- void protocolCopyPaste (uint8_t orientation=1)

  protocolCopyPaste
- void protocolText ()

  protocolText
- void **setup** ()
- void **loop** ()

Variables

- Screen_HX8353E **myScreen**

8.2.1 Detailed Description

Main sketch. Measure the speed of the screen

Developed with embedXcode+

Author

Rei VILO
http://embeddedcomputing.weebly.com

Date

Oct 05, 2013

Version

104

Copyright

See Also

ReadMe.txt for references

8.2.2 Function Documentation

8.2.2.1 void protocolCopyPaste (uint8_t *orientation* = 1)

protocolCopyPaste

measure time to copy-paste a 64x64 area

Parameters

orientation	default=1

8.2.2.2 void protocolSquare (uint16_t *pixels*)

protocolSquare

measure time to draw a square with side=pixels

Parameters

pixels	number of pixels of one side

8.2.2.3 void protocolText ( )

protocolText

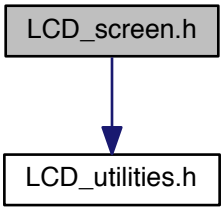
measure time to draw text in 3 fonts, 4 orientations, 10x

8.3 LCD_screen.h File Reference

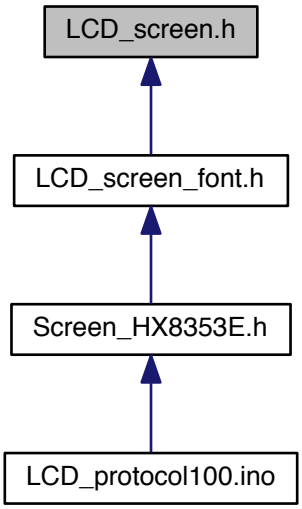
Class library header.

```
#include "LCD_utilities.h"
```

Include dependency graph for LCD_screen.h:



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



Classes

- class LCD_screen

  Generic LCD class.

Macros

- #define LCD_SCREEN_RELEASE 114

Library release number.

Variables

Colours constants

- const uint16_t blackColour = 0b0000000000000000
  black
- const uint16_t whiteColour = 0b1111111111111111
  white
- const uint16_t redColour = 0b1111100000000000
  red
- const uint16_t greenColour = 0b0000011111100000
  green
- const uint16_t blueColour = 0b0000000000011111
  blue
- const uint16_t yellowColour = 0b1111111111100000
  yellow
- const uint16_t cyanColour = 0b0000011111111111
  cyan
- const uint16_t orangeColour = 0b1111101111100000
  orange
- const uint16_t magentaColour = 0b1111100000001111
  magenta
- const uint16_t violetColour = 0b1111100000011111
  violet
- const uint16_t grayColour = 0b0111101111101111
  gray
- const uint16_t darkGrayColour = 0b0011100111100111
  dark gray

8.3.1 Detailed Description

Class library header. Generic LCD class library

Project LCD_screen

Developed with `embedXcode`

Author

Rei VILO
embedXcode.weebly.com

Date

Dec 10, 2013

Version

114

Copyright

See Also

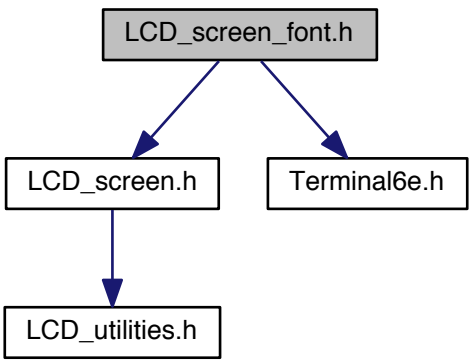
ReadMe.txt for references

8.4 LCD_screen_font.h File Reference

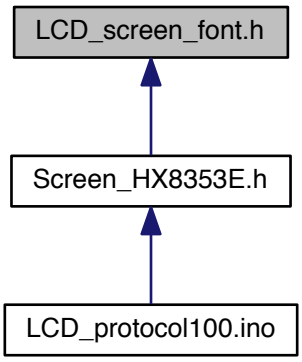
Class library header.

```
#include "LCD_screen.h"
#include "Terminal6e.h"
```

Include dependency graph for LCD_screen_font.h:



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



Classes

- class LCD_screen_font

  Generic LCD with font class.

Macros

- #define LCD_SCREEN_FONT_RELEASE 114

  Library release number.
- #define MAX_FONT_SIZE 1

  Biggest font size.

8.4.1 Detailed Description

Class library header. Generic LCD with font class library

Project LCD_screen_font_main

Developed with `embedXcode`

Author

Rei VILO
embedXcode.weebly.com

Date

Dec 10, 2013

Version

114

Copyright

See Also

ReadMe.txt for references

8.4.2 Macro Definition Documentation

8.4.2.1 #define MAX_FONT_SIZE 1

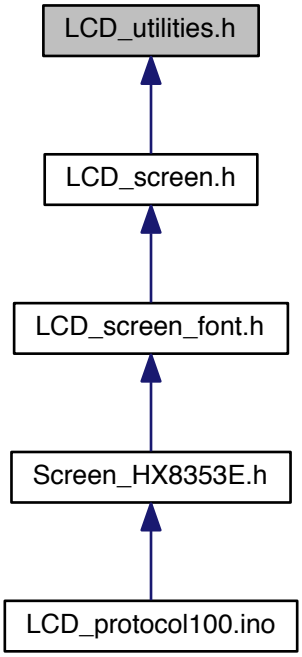
Biggest font size.

Based on the MCU, by default=0

8.5 LCD_utilities.h File Reference

Library header.

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



Macros

- #define LCD_UTILITIES_RELEASE 102
  Library release number.

Functions

Utilities

- int32_t cos32x100 (int32_t degreesX100)
  Cosinus.
- int32_t sin32x100 (int32_t degreesX100)
  Sinus.
- String utf2iso (String s)
  UTF-8 to ISO-8859-1 Converter.

Format

Utilities to format float, 64-bit unsigned integer, hexadecimal and period into string

- String htoa (uint32_t number, uint8_t size=0)
  Convert hexadecimal to string.
- String btoa (uint16_t number, uint8_t size=8)
  Convert binary to string.
- String ttoa (uint32_t number, uint8_t size=0)
  Convert time is ms to string.
- String i32toa (int32_t number, int32_t unit=1, uint8_t decimal=0, uint8_t size=0)
  Convert int32_t to string.

8.5.1 Detailed Description

Library header. Utilities for LCD_screen

Project LCD_screen

Developed with embedXcode

Author

Rei VILO
embedXcode.weebly.com

Date

Sep 18, 2013

Version

102

Copyright

See Also

ReadMe.txt for references

8.5.2 Function Documentation

8.5.2.1 String btoa (uint16_t *number*, uint8_t *size* = 8)

Convert binary to string.

Parameters

number	binary value
size	total number of digits, default=0=no check

Returns

formated string

8.5.2.2 int32_t cos32x100 (int32_t *degreesX100*)

Cosinus.

Parameters

degreesX100	angle in degrees, x100

Returns

cosinus value, x100

Note

This function uses integers only.

8.5.2.3 String htoa (uint32_t *number,* uint8_t *size* = 0)

Convert hexadecimal to string.

Parameters

number	hexadecimal value
size	total number of digits, default=0=no check

Returns

formated string

8.5.2.4 String i32toa (int32_t *number,* int32_t *unit* = 1, uint8_t *decimal* = 0, uint8_t *size* = 0)

Convert int32_t to string.

Parameters

number	value, int32_t, already multiplied by unit
unit	default=1, 10 or 100
decimal	number of decimal digits, default=0
size	total number of digits, default=0=free size, no check

Note

size >= integer digits + 1 for decimal separator . + decimal=decimal digits

Returns

formated string

Note

In case of insufficient place or overflow, # is returned

8.5.2.5 int32_t sin32x100 (int32_t *degreesX100*)

Sinus.

Parameters

degreesX100	angle in degrees, x100

Returns

sinus value, x100

Note

This function uses integers only.

8.5.2.6 String ttoa (uint32_t *number*, uint8_t *size* = 0)

Convert time is ms to string.

Parameters

number	ms
size	total number of digits, default=0=free size, no check

Returns

formated string with time unit, ms, s, mn, h

Note

Automatic selection of the time unit: ms, s, mn, h
In case of insufficient place or overflow, # is returned

8.5.2.7 String utf2iso (String *s*)

UTF-8 to ISO-8859-1 Converter.

Parameters

s	UTF-8 string, input

Returns

ISO-8859-1 string, output

See Also

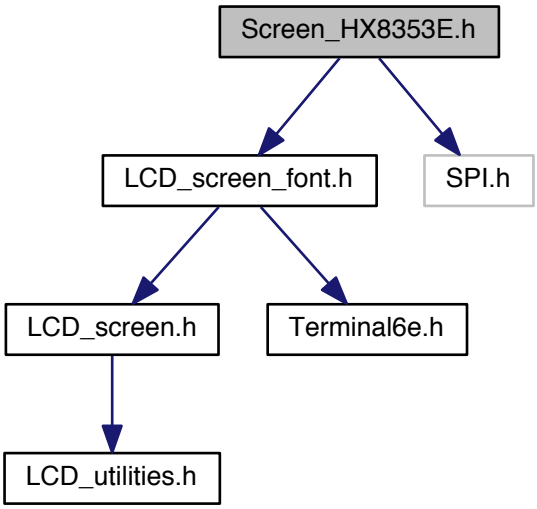
The Unicode Consortium. The Unicode Standard, Version 6.2.0, (Mountain View, CA: The Unicode Consortium, 2012. ISBN 978-1-936213-07-8) http://www.unicode.org/versions/Unicode6.2.0/

8.6 Screen_HX8353E.h File Reference

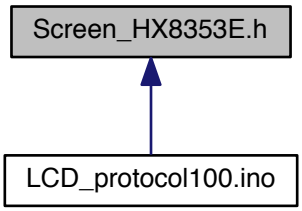
Library header.

```
#include "LCD_screen_font.h"
#include "SPI.h"
```

Include dependency graph for Screen_HX8353E.h:



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



Classes

- class Screen_HX8353E

  Class for Educational BoosterPack MKII.

Macros

- #define SCREEN_HX8353_RELEASE 100

  Library release number.

8.6.1 Detailed Description

Library header. HX8353E screen library

Project new_screen_HX8353

Developed with embedXcode

Author

Rei VILO
embedXcode.weebly.com

Date

Dec 06, 2013

Version

100

Copyright

Dual license:

- For hobbyists and for personal usage: Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)
- For professionals or organisations or for commercial usage: All rights reserved

For any enquiry about license, http://embeddedcomputing.weebly.com/contact

See Also

- Fonts generated with MikroElektronika GLCD Font Creator 1.2.0.0
  http://www.mikroe.com
- LCD_screen Library Suite
  http://embeddedcomputing.weebly.com/lcd_screen-library-suite.html
- Serial_LCD Library Suite
  http://embeddedcomputing.weebly.com/serial-lcd.html

8.7 Terminal12e.h File Reference

Extended font library.

Macros

- #define TERMINAL12E_FONT_RELEASE 102

  Library release number.

8.7.1 Detailed Description

Extended font library. Font Terminal 12 x 16

Developed with embedXcode

Author

Rei VILO
http://embeddedcomputing.weebly.com

Date

May 26, 2012

Version

102

Copyright

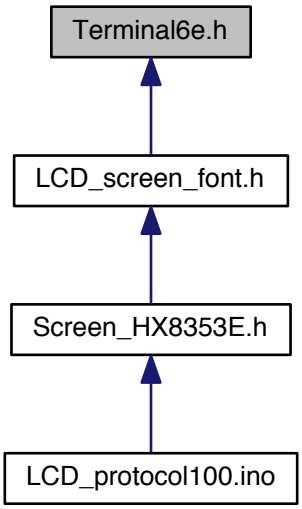
See Also

Font Generated by MikroElektronika GLCD Font Creator 1.2.0.0
MikroeElektronika 2011 http://www.mikroe.com

8.8 Terminal6e.h File Reference

Extended font library.

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



Macros

- #define TERMINAL6E_FONT_RELEASE 102

  Library release number.

8.8.1 Detailed Description

Extended font library. Font Terminal 6 x 8

Developed with embedXcode

Author

Rei VILO
http://embeddedcomputing.weebly.com

Date

May 26, 2012

Version

102

Copyright

See Also

Font Generated by MikroElektronika GLCD Font Creator 1.2.0.0
MikroeElektronika 2011 http://www.mikroe.com

8.9 Terminal8e.h File Reference

Extended font library.

Macros

- #define TERMINAL8E_FONT_RELEASE 102

  Library release number.

8.9.1 Detailed Description

Extended font library. Font Terminal 8 x 12

Developed with embedXcode

Author

Rei VILO
http://embeddedcomputing.weebly.com

Date

May 26, 2012

Version

102

Copyright

See Also

Font Generated by MikroElektronika GLCD Font Creator 1.2.0.0
MikroeElektronika 2011 http://www.mikroe.com