

# Plunger Controller

v.1.0

Generated by Doxygen 1.12.0



<b>1 File Index</b>	<b>1</b>
1.1 File List	1
<b>2 File Documentation</b>	<b>3</b>
2.1 button/button.cpp File Reference	3
2.1.1 Detailed Description	3
2.1.2 Function Documentation	3
2.1.2.1 doublePressHandler()	3
2.1.2.2 longPressHandler()	4
2.1.2.3 singlePressHandler()	4
2.2 button/button.h File Reference	4
2.2.1 Detailed Description	5
2.2.2 Function Documentation	5
2.2.2.1 doublePressHandler()	5
2.2.2.2 longPressHandler()	5
2.2.2.3 singlePressHandler()	6
2.3 button.h	6
2.4 arimo12.h	7
2.5 arimo9.h	9
2.6 custom_fonts.h	11
2.7 dialog12.h	12
2.8 dialog9.h	14
2.9 gothic12.h	16
2.10 gothic9.h	19
2.11 mono12.h	21
2.12 mono9.h	24
2.13 opensans12.h	26
2.14 opensans9.h	28
2.15 roboto12.h	31
2.16 roboto9.h	33
2.17 SansSerif12.h	35
2.18 SansSerif9.h	38
2.19 serif12.h	40
2.20 serif9.h	43
2.21 globals/globals.cpp File Reference	45
2.21.1 Detailed Description	48
2.21.2 Variable Documentation	49
2.21.2.1 backgroundcolors	49
2.21.2.2 backgroundpossibleColors	49
2.21.2.3 centerX	49
2.21.2.4 colormenuArray	49
2.21.2.5 currentMenu	49

2.21.2.6 DIR_PIN	50
2.21.2.7 images_main	50
2.21.2.8 lastClk	50
2.21.2.9 LOOP_TICKS	50
2.21.2.10 menuIndex	50
2.21.2.11 minBrightness	51
2.21.2.12 possibleFont12	51
2.21.2.13 screenOn	51
2.21.2.14 screenUpdateQueue	51
2.21.2.15 selectedBackgroundColor	51
2.22 globals/globals.h File Reference	52
2.22.1 Detailed Description	56
2.22.2 Enumeration Type Documentation	56
2.22.2.1 MenuState	56
2.22.2.2 ScreenUpdateCommand	57
2.22.3 Variable Documentation	57
2.22.3.1 backgroundcolors	57
2.22.3.2 backgroundpossibleColors	57
2.22.3.3 centerX	58
2.22.3.4 colormenuArray	58
2.22.3.5 currentMenu	58
2.22.3.6 DIR_PIN	58
2.22.3.7 images_main	58
2.22.3.8 lastClk	59
2.22.3.9 LOOP_TICKS	59
2.22.3.10 menuIndex	59
2.22.3.11 minBrightness	59
2.22.3.12 possibleFont12	59
2.22.3.13 screenOn	60
2.22.3.14 screenUpdateQueue	60
2.22.3.15 selectedBackgroundColor	60
2.23 globals.h	60
2.24 GUI/GUI.cpp File Reference	62
2.24.1 Detailed Description	63
2.24.2 Function Documentation	63
2.24.2.1 drawBrightnessBar()	63
2.24.2.2 drawDialMarks()	63
2.24.2.3 drawNeedle()	63
2.24.2.4 drawSelectionBar()	64
2.24.2.5 printDateTime()	64
2.24.2.6 setBrightness()	64
2.24.2.7 updateDial()	65

2.25 GUI/GUI.h File Reference . . . . .	65
2.25.1 Detailed Description . . . . .	66
2.25.2 Function Documentation . . . . .	66
2.25.2.1 drawBrightnessBar() . . . . .	66
2.25.2.2 drawDialMarks() . . . . .	66
2.25.2.3 drawNeedle() . . . . .	67
2.25.2.4 drawSelectionBar() . . . . .	68
2.25.2.5 printDateTime() . . . . .	68
2.25.2.6 setBrightness() . . . . .	69
2.25.2.7 updateDial() . . . . .	69
2.26 GUI.h . . . . .	69
2.27 automatic_icon.h . . . . .	70
2.28 brightness_icon.h . . . . .	71
2.29 color_icon.h . . . . .	72
2.30 empty_sun.h . . . . .	73
2.31 ethernet_active.h . . . . .	74
2.32 ethernet_icon.h . . . . .	75
2.33 font_icon.h . . . . .	76
2.34 full_sun.h . . . . .	78
2.35 granasat_logo.h . . . . .	79
2.36 images.h . . . . .	93
2.37 manual_icon.h . . . . .	93
2.38 settings_icon.h . . . . .	94
2.39 step_icon.h . . . . .	96
2.40 libraries.h File Reference . . . . .	97
2.40.1 Detailed Description . . . . .	97
2.41 libraries.h . . . . .	98
2.42 motor/motor.cpp File Reference . . . . .	98
2.42.1 Detailed Description . . . . .	99
2.42.2 Function Documentation . . . . .	99
2.42.2.1 moveMotor() . . . . .	99
2.43 motor/motor.h File Reference . . . . .	99
2.43.1 Detailed Description . . . . .	99
2.43.2 Function Documentation . . . . .	100
2.43.2.1 moveMotor() . . . . .	100
2.44 motor.h . . . . .	101
2.45 project_tasks/projectTasks.cpp File Reference . . . . .	101
2.45.1 Detailed Description . . . . .	102
2.45.2 Function Documentation . . . . .	102
2.45.2.1 motorTask() . . . . .	102
2.45.2.2 rotarymotorTask() . . . . .	102
2.45.2.3 taskButtonPress() . . . . .	102

---

2.45.2.4 taskEncoder()	103
2.45.2.5 taskEthernet()	103
2.45.2.6 taskScreenTimeout()	103
2.45.2.7 taskShowTime()	103
2.45.2.8 taskStartScreen()	104
2.45.2.9 taskUpdateScreen()	104
2.46 project_tasks/projectTasks.h File Reference	104
2.46.1 Detailed Description	105
2.46.2 Function Documentation	105
2.46.2.1 motorTask()	105
2.46.2.2 rotarymotorTask()	105
2.46.2.3 taskButtonPress()	106
2.46.2.4 taskEncoder()	106
2.46.2.5 taskEthernet()	106
2.46.2.6 taskScreenTimeout()	106
2.46.2.7 taskShowTime()	107
2.46.2.8 taskStartScreen()	107
2.46.2.9 taskUpdateScreen()	107
2.47 projectTasks.h	108
2.48 setup/setup.h File Reference	108
2.48.1 Detailed Description	108
2.48.2 Function Documentation	108
2.48.2.1 setup()	108
2.49 setup.h	111

# Chapter 1

## File Index

### 1.1 File List

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

<a href="#">libraries.h</a>	This header file groups all the libraries and external dependencies required for the project. Including this file in the code ensures the correct inclusion of all necessary libraries, simplifying dependency management and keeping the main code clean and organized . . . . .	97
<a href="#">button/button.cpp</a>	Implementation file for button press handling functions . . . . .	3
<a href="#">button/button.h</a>	Header file for button press handling functions . . . . .	4
<a href="#">custom_fonts/arimo12.h</a>	. . . . .	7
<a href="#">custom_fonts/arimo9.h</a>	. . . . .	9
<a href="#">custom_fonts/custom_fonts.h</a>	. . . . .	11
<a href="#">custom_fonts/dialog12.h</a>	. . . . .	12
<a href="#">custom_fonts/dialog9.h</a>	. . . . .	14
<a href="#">custom_fonts/gothic12.h</a>	. . . . .	16
<a href="#">custom_fonts/gothic9.h</a>	. . . . .	19
<a href="#">custom_fonts/mono12.h</a>	. . . . .	21
<a href="#">custom_fonts/mono9.h</a>	. . . . .	24
<a href="#">custom_fonts/opensans12.h</a>	. . . . .	26
<a href="#">custom_fonts/opensans9.h</a>	. . . . .	28
<a href="#">custom_fonts/roboto12.h</a>	. . . . .	31
<a href="#">custom_fonts/roboto9.h</a>	. . . . .	33
<a href="#">custom_fonts/SansSerif12.h</a>	. . . . .	35
<a href="#">custom_fonts/SansSerif9.h</a>	. . . . .	38
<a href="#">custom_fonts/serif12.h</a>	. . . . .	40
<a href="#">custom_fonts/serif9.h</a>	. . . . .	43
<a href="#">globals/globals.cpp</a>	This file contains global variables, constants, and configurations used throughout the project . . . . .	45
<a href="#">globals/globals.h</a>	Global variable and constant definitions for the project . . . . .	52
<a href="#">GUI/GUI.cpp</a>	Implementation of the functions related to the graphical user interface (GUI) . . . . .	62
<a href="#">GUI/GUI.h</a>	Graphical User Interface (GUI) module for managing the display elements . . . . .	65
<a href="#">icons/automatic_icon.h</a>	. . . . .	70
<a href="#">icons/brightness_icon.h</a>	. . . . .	71

icons/color_icon.h . . . . .	72
icons/empty_sun.h . . . . .	73
icons/ethernet_active.h . . . . .	74
icons/ethernet_icon.h . . . . .	75
icons/font_icon.h . . . . .	76
icons/full_sun.h . . . . .	78
icons/granasat_logo.h . . . . .	79
icons/images.h . . . . .	93
icons/manual_icon.h . . . . .	93
icons/settings_icon.h . . . . .	94
icons/step_icon.h . . . . .	96
motor/motor.cpp . . . . .	
Implementation file for motor control functions . . . . .	98
motor/motor.h . . . . .	
Header file for motor control functions . . . . .	99
project_tasks/projectTasks.cpp . . . . .	
Header file for system task management . . . . .	101
project_tasks/projectTasks.h . . . . .	
Header file for system task management . . . . .	104
setup/setup.h . . . . .	
Contains all setup configurations for the project including Ethernet, LCD, Motor, Pin, RTC, and Task setups . . . . .	108



## Chapter 2

# File Documentation

### 2.1 button/button.cpp File Reference

Implementation file for button press handling functions.

```
#include "button.h"
#include <globals/globals.h>
#include <motor/motor.h>
#include <GUI/GUI.h>
```

#### Functions

- void [singlePressHandler](#) (BfButton \*btn, BfButton::press\_pattern\_t pattern)  
*Handles single button press events.*
- void [doublePressHandler](#) (BfButton \*btn, BfButton::press\_pattern\_t pattern)  
*Handles double button press events.*
- void [longPressHandler](#) (BfButton \*btn, BfButton::press\_pattern\_t pattern)  
*Handles long button press events.*

#### 2.1.1 Detailed Description

Implementation file for button press handling functions.

This file contains the implementation of functions that handle button press events, including single, double, and long presses. These functions allow users to navigate through menus, control the motor, adjust screen settings, and perform other actions.

#### 2.1.2 Function Documentation

##### 2.1.2.1 doublePressHandler()

```
void doublePressHandler (
    BfButton * btn,
    BfButton::press_pattern_t pattern)
```

Handles double button press events.

This function toggles the engineering mode when a double press is detected.

**Parameters**

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (double press in this case).

**2.1.2.2 longPressHandler()**

```
void longPressHandler (
    BfButton * btn,
    BfButton::press_pattern_t pattern)
```

Handles long button press events.

This function toggles the screen on/off. If the screen is turned off, it reduces the brightness to zero and puts the display into sleep mode. When turned back on, the display is woken up and the brightness is restored to the previous level.

**Parameters**

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (long press in this case).

**2.1.2.3 singlePressHandler()**

```
void singlePressHandler (
    BfButton * btn,
    BfButton::press_pattern_t pattern)
```

Handles single button press events.

This function manages the navigation between different menus (main menu, settings, motor control, etc.) based on the current state and selection. It sends update commands to the screen to reflect the changes in the UI.

**Parameters**

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (single press in this case).

**2.2 button/button.h File Reference**

Header file for button press handling functions.

```
#include <libraries.h>
```

## Functions

- void [singlePressHandler](#) (BfButton \*btn, BfButton::press\_pattern\_t pattern)  
*Handles single button press events.*
- void [doublePressHandler](#) (BfButton \*btn, BfButton::press\_pattern\_t pattern)  
*Handles double button press events.*
- void [longPressHandler](#) (BfButton \*btn, BfButton::press\_pattern\_t pattern)  
*Handles long button press events.*

### 2.2.1 Detailed Description

Header file for button press handling functions.

This file contains the declarations of the button press handling functions, including single, double, and long press actions for navigating the menu and controlling various settings.

### 2.2.2 Function Documentation

#### 2.2.2.1 doublePressHandler()

```
void doublePressHandler (  
    BfButton * btn,  
    BfButton::press_pattern_t pattern)
```

Handles double button press events.

This function is called when a double press is detected. It toggles the engineering mode or any other logic that needs to be implemented for a double press event.

##### Parameters

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (double press in this case).

This function toggles the engineering mode when a double press is detected.

##### Parameters

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (double press in this case).

#### 2.2.2.2 longPressHandler()

```
void longPressHandler (  
    BfButton * btn,  
    BfButton::press_pattern_t pattern)
```

Handles long button press events.

This function is called when a long press is detected. It toggles the screen on/off, adjusts the brightness, and manages the sleep mode of the display.

**Parameters**

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (long press in this case).

This function toggles the screen on/off. If the screen is turned off, it reduces the brightness to zero and puts the display into sleep mode. When turned back on, the display is woken up and the brightness is restored to the previous level.

**Parameters**

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (long press in this case).

**2.2.2.3 singlePressHandler()**

```
void singlePressHandler (
    BfButton * btn,
    BfButton::press_pattern_t pattern)
```

Handles single button press events.

This function is called when a single press is detected. It manages the navigation through different menus and performs specific actions based on the current menu and selection.

**Parameters**

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (single press in this case).

This function manages the navigation between different menus (main menu, settings, motor control, etc.) based on the current state and selection. It sends update commands to the screen to reflect the changes in the UI.

**Parameters**

<i>btn</i>	Pointer to the button object.
<i>pattern</i>	The press pattern detected (single press in this case).

**2.3 button.h**

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

```
00001
00010 #ifndef BUTTON_H
00011 #define BUTTON_H
00012 #include <libraries.h>
00013
00024 void singlePressHandler(BfButton *btn, BfButton::press_pattern_t pattern);
00025
00036 void doublePressHandler(BfButton *btn, BfButton::press_pattern_t pattern);
00037
00047 void longPressHandler(BfButton *btn, BfButton::press_pattern_t pattern);
00048
00049 #endif // BUTTON_H
```

## 2.4 arimo12.h

```

00001 // Created by https://oledisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Arimo_Regular_l2Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0xA8, 0x80, // '!'
00008     0x52, 0x94, // '"'
00009     0x24, 0x24, 0xFE, 0x24, 0x28, 0x48, 0xFE, 0x48, 0x48, // '#'
00010     0x10, 0x7C, 0x54, 0x50, 0x70, 0x3C, 0x16, 0x12, 0xD6, 0x7C, 0x10, // '$'
00011     0x71, 0x09, 0x10, 0x92, 0x09, 0x20, 0x75, 0xC0, 0x92, 0x09, 0x21, 0x12, 0x11, 0xC0, // '%'
00012     0x18, 0x12, 0x09, 0x07, 0x07, 0x22, 0x92, 0x30, 0x88, 0x7B, 0x00, // '&'
00013     0xA8, // "'"
00014     0x44, 0x88, 0x88, 0x88, 0x84, 0x60, // '('
00015     0x44, 0x22, 0x22, 0x22, 0x24, 0x40, // ')'
00016     0x23, 0xE2, 0x14, // '*'
00017     0x20, 0x8F, 0x88, 0x20, // '+'
00018     0xA8, // ','
00019     0xC0, // '-'
00020     0x80, // '.'
00021     0x22, 0x44, 0x44, 0x48, 0x80, // '/'
00022     0x38, 0x44, 0x44, 0x44, 0xC6, 0x44, 0x44, 0x38, // '0'
00023     0x62, 0x82, 0x08, 0x20, 0x82, 0x08, 0xF8, // '1'
00024     0x72, 0x20, 0x82, 0x10, 0x84, 0x20, 0xF8, // '2'
00025     0x38, 0x88, 0x10, 0x21, 0x80, 0x81, 0x22, 0x38, // '3'
00026     0x08, 0x18, 0x18, 0x28, 0x48, 0x48, 0xFE, 0x08, 0x08, // '4'
00027     0x7C, 0x81, 0x03, 0xC0, 0x40, 0x81, 0x22, 0x78, // '5'
00028     0x72, 0x28, 0x3C, 0x8A, 0x28, 0xA2, 0x70, // '6'
00029     0xF8, 0x21, 0x04, 0x20, 0x82, 0x10, 0x40, // '7'
00030     0x72, 0x28, 0xA2, 0x72, 0x28, 0xA2, 0x70, // '8'
00031     0x72, 0x28, 0xA2, 0x89, 0xE0, 0xA2, 0x70, // '9'
00032     0x80, 0x08, // ':'
00033     0x80, 0x0A, 0x80, // ';'
00034     0x09, 0xC8, 0x1C, 0x08, // '<'
00035     0xF8, 0x00, 0x3E, // '='
00036     0x81, 0xC0, 0x9C, 0x80, // '>'
00037     0x71, 0x10, 0x30, 0x41, 0x86, 0x08, 0x00, 0x20, // '?'
00038     0x1F, 0x04, 0x11, 0x35, 0x49, 0xAA, 0x25, 0x44, 0xA8, 0x94, 0xFC, 0x80, 0x08, 0x20, 0xF8, 0x00, // '@'
00039     0x10, 0x28, 0x28, 0x28, 0x44, 0x7C, 0x44, 0xC6, 0x82, // 'A'
00040     0xF9, 0x0A, 0x14, 0x6F, 0x90, 0xA1, 0x42, 0xF8, // 'B'
00041     0x3C, 0x61, 0x20, 0x10, 0x08, 0x04, 0x02, 0x00, 0x84, 0x3C, 0x00, // 'C'
00042     0xF8, 0x86, 0x82, 0x82, 0x82, 0x82, 0x82, 0x84, 0xF8, // 'D'
00043     0xFD, 0x02, 0x04, 0x0F, 0xD0, 0x20, 0x40, 0xFC, // 'E'
00044     0xFD, 0x02, 0x04, 0x0F, 0x90, 0x20, 0x40, 0x80, // 'F'
00045     0x7C, 0xC6, 0x80, 0x80, 0x8E, 0x82, 0x82, 0xC6, 0x3C, // 'G'
00046     0x82, 0x82, 0x82, 0x82, 0xFE, 0x82, 0x82, 0x82, 0x82, // 'H'
00047     0xAA, 0xAA, 0x80, // 'I'
00048     0x38, 0x20, 0x82, 0x08, 0x20, 0xA2, 0x70, // 'J'
00049     0x84, 0x88, 0x90, 0xA0, 0xE0, 0x90, 0x88, 0x8C, 0x84, // 'K'
00050     0x81, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0xFC, // 'L'
00051     0xC2, 0xC6, 0xC6, 0xC6, 0xAA, 0xAA, 0xAA, 0x92, 0x92, // 'M'
00052     0xC2, 0xC2, 0xA2, 0xA2, 0x92, 0x8A, 0x8A, 0x86, 0x86, // 'N'
00053     0x7C, 0xC6, 0x82, 0x82, 0x82, 0x82, 0x82, 0xC6, 0x7C, // 'O'
00054     0xF9, 0x0A, 0x14, 0x28, 0x5F, 0x20, 0x40, 0x80, // 'P'
00055     0x7C, 0xC6, 0x82, 0x82, 0x82, 0x82, 0x82, 0xC6, 0x7C, 0x18, 0x0C, // 'Q'
00056     0xFC, 0x82, 0x82, 0x82, 0xFC, 0x88, 0x84, 0x86, 0x82, // 'R'
00057     0x79, 0x0A, 0x07, 0x01, 0xC0, 0x81, 0x42, 0x78, // 'S'
00058     0xFE, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, // 'T'
00059     0x82, 0x82, 0x82, 0x82, 0x82, 0x82, 0x82, 0xC6, 0x7C, // 'U'
00060     0x82, 0x44, 0x44, 0x44, 0x6C, 0x28, 0x28, 0x28, 0x10, // 'V'
00061     0x84, 0x2C, 0x46, 0x4A, 0x44, 0x4A, 0x4A, 0x46, 0xAC, 0x31, 0x83, 0x18, 0x31, 0x80, // 'W'
00062     0x44, 0x44, 0x28, 0x38, 0x10, 0x28, 0x28, 0x44, 0x44, // 'X'
00063     0x44, 0x44, 0x28, 0x28, 0x10, 0x10, 0x10, 0x10, // 'Y'
00064     0x7C, 0x04, 0x08, 0x18, 0x10, 0x20, 0x60, 0x40, 0xFE, // 'Z'
00065     0xD2, 0x49, 0x24, 0x93, 0x00, // '['
00066     0x88, 0x44, 0x44, 0x42, 0x20, // '\'
00067     0xC9, 0x24, 0x92, 0x4B, 0x00, // ']'
00068     0x21, 0x45, 0x22, // '^'
00069     0xFE, // '_'
00070     0x88, // '`'
00071     0x71, 0x10, 0x27, 0xC8, 0x93, 0x3B, 0x00, // 'a'
00072     0x82, 0x0F, 0xA2, 0x8A, 0x28, 0xA2, 0xF8, // 'b'
00073     0x72, 0x68, 0x20, 0x82, 0x67, 0x00, // 'c'
00074     0x08, 0x2F, 0xA2, 0x8A, 0x28, 0xA2, 0xF8, // 'd'
00075     0x72, 0x28, 0xBE, 0x82, 0x27, 0x00, // 'e'
00076     0x64, 0xE4, 0x44, 0x44, 0x40, // 'f'
00077     0xFA, 0x28, 0xA2, 0x8A, 0x2F, 0x82, 0xF0, // 'g'
00078     0x82, 0x0F, 0xA2, 0x8A, 0x28, 0xA2, 0x88, // 'h'
00079     0x8A, 0xAA, 0x80, // 'i'
00080     0x41, 0x24, 0x92, 0x4B, 0x00, // 'j'
00081     0x82, 0x09, 0x2C, 0xA3, 0x8A, 0x24, 0x98, // 'k'
00082     0xAA, 0xAA, 0x80, // 'l'
00083     0xFF, 0xA2, 0x28, 0x8A, 0x22, 0x88, 0xA2, 0x28, 0x88, // 'm'
00084     0xFA, 0x28, 0xA2, 0x8A, 0x28, 0x80, // 'n'
00085     0x72, 0x28, 0xA2, 0x8A, 0x27, 0x00, // 'o'

```

```

00086     0xFA, 0x28, 0xA2, 0x8A, 0x2F, 0xA0, 0x80, // 'p'
00087     0xFA, 0x28, 0xA2, 0x8A, 0x2F, 0x82, 0x08, // 'q'
00088     0xE8, 0x88, 0x88, 0x80, // 'r'
00089     0x7C, 0x44, 0x40, 0x3C, 0x04, 0x84, 0x7C, // 's'
00090     0x44, 0xE4, 0x44, 0x44, 0x60, // 't'
00091     0x8A, 0x28, 0xA2, 0x8A, 0x2F, 0x80, // 'u'
00092     0x8A, 0x25, 0x14, 0x51, 0x42, 0x00, // 'v'
00093     0x88, 0xA5, 0x25, 0x51, 0x54, 0x55, 0x15, 0x42, 0x20, // 'w'
00094     0xD9, 0x45, 0x08, 0x51, 0x49, 0x80, // 'x'
00095     0x8B, 0x25, 0x14, 0x50, 0x82, 0x08, 0xC0, // 'y'
00096     0x70, 0x43, 0x08, 0x61, 0x0F, 0x80, // 'z'
00097     0x32, 0x10, 0x84, 0x61, 0x08, 0x42, 0x0C, // '{'
00098     0xAA, 0xAA, 0xAA, // '|'
00099     0xC1, 0x08, 0x42, 0x18, 0x84, 0x23, 0x30 // '}'
00100 };
00101 const GFXglyph Arimo_Regular_12Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103     { 0, 2, 1, 4, 0, -1 }, // ' '
00104     { 1, 2, 9, 4, 1, -9 }, // '!'
00105     { 4, 5, 3, 5, 0, -9 }, // '"'
00106     { 6, 8, 9, 8, 0, -9 }, // '#'
00107     { 15, 8, 11, 8, 0, -10 }, // '$'
00108     { 26, 12, 9, 12, 0, -9 }, // '%'
00109     { 40, 9, 9, 9, 0, -9 }, // '&'
00110     { 51, 2, 3, 3, 1, -9 }, // "'"
00111     { 52, 4, 11, 5, 1, -9 }, // '('
00112     { 58, 4, 11, 5, 0, -9 }, // ')'
00113     { 64, 6, 4, 6, 0, -9 }, // '*'
00114     { 67, 6, 5, 8, 1, -7 }, // '+'
00115     { 71, 2, 3, 4, 1, -1 }, // ','
00116     { 72, 3, 1, 5, 1, -4 }, // '-'
00117     { 73, 2, 1, 4, 1, -1 }, // '.'
00118     { 74, 4, 9, 4, 0, -9 }, // '/'
00119     { 79, 8, 9, 8, 0, -9 }, // '0'
00120     { 88, 6, 9, 8, 1, -9 }, // '1'
00121     { 95, 6, 9, 8, 1, -9 }, // '2'
00122     { 102, 7, 9, 8, 0, -9 }, // '3'
00123     { 110, 8, 9, 8, 0, -9 }, // '4'
00124     { 119, 7, 9, 8, 0, -9 }, // '5'
00125     { 127, 6, 9, 8, 1, -9 }, // '6'
00126     { 134, 6, 9, 8, 1, -9 }, // '7'
00127     { 141, 6, 9, 8, 1, -9 }, // '8'
00128     { 148, 6, 9, 8, 1, -9 }, // '9'
00129     { 155, 2, 7, 4, 1, -7 }, // ':'
00130     { 157, 2, 9, 4, 1, -7 }, // ';'
00131     { 160, 6, 5, 8, 1, -7 }, // '<'
00132     { 164, 6, 4, 8, 1, -6 }, // '='
00133     { 167, 6, 5, 8, 1, -7 }, // '>'
00134     { 171, 7, 9, 8, 1, -9 }, // '?'
00135     { 179, 11, 11, 13, 1, -9 }, // '@'
00136     { 195, 8, 9, 8, 0, -9 }, // 'A'
00137     { 204, 7, 9, 9, 1, -9 }, // 'B'
00138     { 212, 9, 9, 10, 1, -9 }, // 'C'
00139     { 223, 8, 9, 10, 1, -9 }, // 'D'
00140     { 232, 7, 9, 9, 1, -9 }, // 'E'
00141     { 240, 7, 9, 8, 1, -9 }, // 'F'
00142     { 248, 8, 9, 10, 1, -9 }, // 'G'
00143     { 257, 8, 9, 10, 1, -9 }, // 'H'
00144     { 266, 2, 9, 4, 1, -9 }, // 'I'
00145     { 269, 6, 9, 7, 0, -9 }, // 'J'
00146     { 276, 8, 9, 9, 1, -9 }, // 'K'
00147     { 285, 7, 9, 8, 1, -9 }, // 'L'
00148     { 293, 8, 9, 10, 1, -9 }, // 'M'
00149     { 302, 8, 9, 10, 1, -9 }, // 'N'
00150     { 311, 8, 9, 10, 1, -9 }, // 'O'
00151     { 320, 7, 9, 9, 1, -9 }, // 'P'
00152     { 328, 8, 11, 10, 1, -9 }, // 'Q'
00153     { 339, 8, 9, 10, 1, -9 }, // 'R'
00154     { 348, 7, 9, 9, 1, -9 }, // 'S'
00155     { 356, 8, 9, 8, 0, -9 }, // 'T'
00156     { 365, 8, 9, 10, 1, -9 }, // 'U'
00157     { 374, 8, 9, 8, 0, -9 }, // 'V'
00158     { 383, 12, 9, 12, 0, -9 }, // 'W'
00159     { 397, 8, 9, 8, 0, -9 }, // 'X'
00160     { 406, 8, 9, 8, 0, -9 }, // 'Y'
00161     { 415, 8, 9, 8, 0, -9 }, // 'Z'
00162     { 424, 3, 11, 4, 1, -9 }, // '['
00163     { 429, 4, 9, 4, 0, -9 }, // '\'
00164     { 434, 3, 11, 4, 0, -9 }, // ']'
00165     { 439, 6, 4, 6, 0, -9 }, // '^'
00166     { 442, 8, 1, 8, 0, 2 }, // '_'
00167     { 443, 3, 2, 5, 1, -10 }, // '`'
00168     { 444, 7, 7, 8, 1, -7 }, // 'a'
00169     { 451, 6, 9, 8, 1, -9 }, // 'b'
00170     { 458, 6, 7, 7, 1, -7 }, // 'c'
00171     { 464, 6, 9, 8, 1, -9 }, // 'd'
00172     { 471, 6, 7, 8, 1, -7 }, // 'e'

```

```

00173     { 477, 4, 9, 4, 0, -9 }, // 'f'
00174     { 482, 6, 9, 8, 1, -7 }, // 'g'
00175     { 489, 6, 9, 8, 1, -9 }, // 'h'
00176     { 496, 2, 9, 4, 1, -9 }, // 'i'
00177     { 499, 3, 11, 4, 0, -9 }, // 'j'
00178     { 504, 6, 9, 7, 1, -9 }, // 'k'
00179     { 511, 2, 9, 4, 1, -9 }, // 'l'
00180     { 514, 10, 7, 12, 1, -7 }, // 'm'
00181     { 523, 6, 7, 8, 1, -7 }, // 'n'
00182     { 529, 6, 7, 8, 1, -7 }, // 'o'
00183     { 535, 6, 9, 8, 1, -7 }, // 'p'
00184     { 542, 6, 9, 8, 1, -7 }, // 'q'
00185     { 549, 4, 7, 5, 1, -7 }, // 'r'
00186     { 553, 8, 7, 8, 0, -7 }, // 's'
00187     { 560, 4, 9, 4, 0, -9 }, // 't'
00188     { 565, 6, 7, 8, 1, -7 }, // 'u'
00189     { 571, 6, 7, 6, 0, -7 }, // 'v'
00190     { 577, 10, 7, 10, 0, -7 }, // 'w'
00191     { 586, 6, 7, 6, 0, -7 }, // 'x'
00192     { 592, 6, 9, 6, 0, -7 }, // 'y'
00193     { 599, 6, 7, 6, 0, -7 }, // 'z'
00194     { 605, 5, 11, 5, 0, -9 }, // '{'
00195     { 612, 2, 12, 4, 1, -9 }, // '|'
00196     { 615, 5, 11, 5, 0, -9 } // '}'
00197 };
00198 const GFXfont Arimo_Regular_12 PROGMEM = {
00199 (uint8_t *)Arimo_Regular_12Bitmaps, (GFXglyph *)Arimo_Regular_12Glyphs, 0x20, 0x7E, 14};

```

## 2.5 arimo9.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Arimo_Regular_9Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0x88, // '!'
00008     0xAA, // '"'
00009     0x01, 0x4F, 0x94, 0xF9, 0x40, 0x00, // '#'
00010     0x21, 0xCA, 0x38, 0x30, 0xA2, 0x9C, 0x20, // '$'
00011     0x64, 0x52, 0x2A, 0x0E, 0xC1, 0x51, 0x28, 0x98, // '%'
00012     0x30, 0x91, 0x43, 0x49, 0x93, 0x1F, 0x00, // '&'
00013     0xA0, // '''
00014     0x52, 0x49, 0x24, 0x40, // '('
00015     0x89, 0x24, 0x92, 0x80, // ')'
00016     0x45, 0x99, 0x20, // '*'
00017     0x20, 0x8F, 0x88, 0x20, // '+'
00018     0xA0, // ','
00019     0xE0, // '-'
00020     0x80, // '.'
00021     0x22, 0x44, 0x48, 0x80, // '/'
00022     0x72, 0x48, 0xA2, 0x8A, 0x47, 0x00, // '0'
00023     0x46, 0x10, 0x84, 0x23, 0xC0, // '1'
00024     0x70, 0x21, 0x04, 0x21, 0x0F, 0x80, // '2'
00025     0x72, 0x21, 0x0C, 0x0A, 0x27, 0x00, // '3'
00026     0x10, 0xC5, 0x14, 0x93, 0xE1, 0x00, // '4'
00027     0x71, 0x07, 0x02, 0x0A, 0x27, 0x00, // '5'
00028     0x71, 0x07, 0x22, 0x89, 0x27, 0x00, // '6'
00029     0xF8, 0x41, 0x08, 0x20, 0x82, 0x00, // '7'
00030     0x72, 0x25, 0x1C, 0x8A, 0x27, 0x00, // '8'
00031     0x72, 0x48, 0xA2, 0x78, 0x47, 0x00, // '9'
00032     0x80, 0x80, // ':'
00033     0x80, 0xA0, // ';'
00034     0x19, 0x88, 0x18, 0x18, // '<'
00035     0xF8, 0x0F, 0x80, // '='
00036     0xC0, 0xC0, 0x8C, 0xC0, // '>'
00037     0x72, 0x20, 0x84, 0x20, 0x02, 0x00, // '?'
00038     0x3C, 0x42, 0xBE, 0xCA, 0xCA, 0xBC, 0x80, 0x78, // '@'
00039     0x30, 0x60, 0xA2, 0x47, 0x90, 0xA1, 0x00, // 'A'
00040     0xF2, 0x49, 0x3C, 0x8A, 0x2F, 0x00, // 'B'
00041     0x3C, 0x44, 0x40, 0x80, 0x40, 0x42, 0x3C, // 'C'
00042     0xF1, 0x12, 0x14, 0x28, 0x91, 0x3C, 0x00, // 'D'
00043     0xF2, 0x08, 0x3C, 0x82, 0x0F, 0x80, // 'E'
00044     0xFA, 0x08, 0x3C, 0x82, 0x08, 0x00, // 'F'
00045     0x38, 0x8A, 0x04, 0x68, 0x48, 0x8E, 0x00, // 'G'
00046     0x8A, 0x28, 0xBE, 0x8A, 0x28, 0x80, // 'H'
00047     0xAA, 0xA8, // 'I'
00048     0x30, 0x84, 0x21, 0x49, 0xC0, // 'J'
00049     0x92, 0x8C, 0x30, 0xA2, 0x49, 0x80, // 'K'
00050     0x84, 0x21, 0x08, 0x43, 0xC0, // 'L'
00051     0x8A, 0x2D, 0xB6, 0xDA, 0xAA, 0x80, // 'M'
00052     0x8B, 0x2C, 0xAA, 0x9A, 0x68, 0x80, // 'N'
00053     0x38, 0x44, 0x82, 0x82, 0x82, 0x44, 0x38, // 'O'

```

```

00054      0xF2, 0x48, 0xA4, 0xE2, 0x08, 0x00, // 'P'
00055      0x38, 0x44, 0x82, 0x82, 0x44, 0x38, 0x10, 0x0C, // 'Q'
00056      0xF9, 0x12, 0x27, 0x89, 0x11, 0x22, 0x00, // 'R'
00057      0x78, 0x81, 0x01, 0xC0, 0x50, 0x9E, 0x00, // 'S'
00058      0xF8, 0x82, 0x08, 0x20, 0x82, 0x00, // 'T'
00059      0x8A, 0x28, 0xA2, 0x8A, 0x27, 0x00, // 'U'
00060      0x84, 0x91, 0x22, 0x43, 0x06, 0x0C, 0x00, // 'V'
00061      0x88, 0xA2, 0x25, 0x51, 0x54, 0x55, 0x15, 0x42, 0x20, // 'W'
00062      0x51, 0x42, 0x08, 0x51, 0x48, 0x80, // 'X'
00063      0x44, 0x44, 0x28, 0x10, 0x10, 0x10, 0x10, // 'Y'
00064      0x78, 0x10, 0x41, 0x02, 0x08, 0x3F, 0x00, // 'Z'
00065      0xD2, 0x49, 0x24, 0xC0, // '['
00066      0x88, 0x44, 0x42, 0x20, // '\'
00067      0xC9, 0x24, 0x92, 0xC0, // ']'
00068      0x4A, 0xA0, // '^'
00069      0xF8, // '_'
00070      0x48, // '`'
00071      0x70, 0x47, 0x24, 0x68, // 'a'
00072      0x84, 0x39, 0x29, 0x4B, 0x80, // 'b'
00073      0x72, 0x08, 0x20, 0x70, // 'c'
00074      0x10, 0x9D, 0x29, 0x49, 0xC0, // 'd'
00075      0x72, 0x2F, 0xA0, 0x70, // 'e'
00076      0x32, 0x3C, 0x84, 0x21, 0x00, // 'f'
00077      0x74, 0xA5, 0x27, 0x09, 0xC0, // 'g'
00078      0x88, 0xEA, 0xAA, 0xA0, // 'h'
00079      0x0A, 0xA8, // 'i'
00080      0x0A, 0xAA, 0x80, // 'j'
00081      0x84, 0x29, 0x8C, 0x52, 0x40, // 'k'
00082      0xAA, 0xA8, // 'l'
00083      0xED, 0x2A, 0x54, 0xA9, 0x40, // 'm'
00084      0xEA, 0xAA, 0xA0, // 'n'
00085      0x72, 0x28, 0xA2, 0x70, // 'o'
00086      0xE4, 0xA5, 0x2E, 0x42, 0x00, // 'p'
00087      0x74, 0xA5, 0x27, 0x08, 0x40, // 'q'
00088      0xD2, 0x48, // 'r'
00089      0x72, 0x07, 0x02, 0x70, // 's'
00090      0x4E, 0x44, 0x46, // 't'
00091      0xAA, 0xAA, 0xE0, // 'u'
00092      0x84, 0x91, 0x21, 0x83, 0x00, // 'v'
00093      0xB5, 0x6A, 0xD2, 0xC4, 0x80, // 'w'
00094      0x51, 0x42, 0x14, 0x88, // 'x'
00095      0x84, 0x91, 0x21, 0x83, 0x04, 0x30, 0x00, // 'y'
00096      0xF1, 0x10, 0x8F, 0x00, // 'z'
00097      0x64, 0x44, 0x84, 0x44, 0x60, // '{'
00098      0xAA, 0xAA, 0x80, // '|'
00099      0xC4, 0x44, 0x24, 0x44, 0xC0 // '}'
00100  };
00101  const GFXglyph Arimo-Regular_9Glyphs[] PROGMEM = {
00102  // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103      { 0, 2, 1, 4, 0, -1 }, // ' '
00104      { 1, 2, 7, 4, 1, -7 }, // '!'
00105      { 3, 4, 2, 4, 0, -7 }, // '"'
00106      { 4, 6, 7, 6, 0, -7 }, // '#'
00107      { 10, 6, 9, 6, 0, -8 }, // '$'
00108      { 17, 9, 7, 9, 0, -7 }, // '%'
00109      { 25, 7, 7, 7, 0, -7 }, // '&'
00110      { 32, 2, 2, 3, 0, -7 }, // "'"
00111      { 33, 3, 9, 4, 1, -7 }, // '('
00112      { 37, 3, 9, 4, 0, -7 }, // ')'
00113      { 41, 5, 4, 5, 0, -7 }, // '*'
00114      { 44, 6, 5, 6, 0, -6 }, // '+'
00115      { 48, 2, 2, 4, 1, -1 }, // ','
00116      { 49, 4, 1, 4, 0, -3 }, // '-'
00117      { 50, 2, 1, 4, 1, -1 }, // '.'
00118      { 51, 4, 7, 4, 0, -7 }, // '/'
00119      { 55, 6, 7, 6, 0, -7 }, // '0'
00120      { 61, 5, 7, 6, 1, -7 }, // '1'
00121      { 66, 6, 7, 6, 0, -7 }, // '2'
00122      { 72, 6, 7, 6, 0, -7 }, // '3'
00123      { 78, 6, 7, 6, 0, -7 }, // '4'
00124      { 84, 6, 7, 6, 0, -7 }, // '5'
00125      { 90, 6, 7, 6, 0, -7 }, // '6'
00126      { 96, 6, 7, 6, 0, -7 }, // '7'
00127      { 102, 6, 7, 6, 0, -7 }, // '8'
00128      { 108, 6, 7, 6, 0, -7 }, // '9'
00129      { 114, 2, 5, 4, 1, -5 }, // ':'
00130      { 116, 2, 6, 4, 1, -5 }, // ';'
00131      { 118, 6, 5, 6, 0, -6 }, // '<'
00132      { 122, 6, 3, 6, 0, -5 }, // '='
00133      { 125, 6, 5, 6, 0, -6 }, // '>'
00134      { 129, 6, 7, 6, 0, -7 }, // '?'
00135      { 135, 8, 8, 10, 1, -7 }, // '@'
00136      { 143, 7, 7, 7, 0, -7 }, // 'A'
00137      { 150, 6, 7, 7, 1, -7 }, // 'B'
00138      { 156, 8, 7, 8, 0, -7 }, // 'C'
00139      { 163, 7, 7, 8, 1, -7 }, // 'D'
00140      { 170, 6, 7, 7, 1, -7 }, // 'E'

```



```

00141     { 176, 6, 7, 7, 1, -7 }, // 'F'
00142     { 182, 7, 7, 8, 0, -7 }, // 'G'
00143     { 189, 6, 7, 8, 1, -7 }, // 'H'
00144     { 195, 2, 7, 4, 1, -7 }, // 'I'
00145     { 197, 5, 7, 6, 0, -7 }, // 'J'
00146     { 202, 6, 7, 7, 1, -7 }, // 'K'
00147     { 208, 5, 7, 6, 1, -7 }, // 'L'
00148     { 213, 6, 7, 8, 1, -7 }, // 'M'
00149     { 219, 6, 7, 8, 1, -7 }, // 'N'
00150     { 225, 8, 7, 8, 0, -7 }, // 'O'
00151     { 232, 6, 7, 7, 1, -7 }, // 'P'
00152     { 238, 8, 9, 8, 0, -7 }, // 'Q'
00153     { 247, 7, 7, 8, 1, -7 }, // 'R'
00154     { 254, 7, 7, 7, 0, -7 }, // 'S'
00155     { 261, 6, 7, 6, 0, -7 }, // 'T'
00156     { 267, 6, 7, 8, 1, -7 }, // 'U'
00157     { 273, 7, 7, 7, 0, -7 }, // 'V'
00158     { 280, 10, 7, 10, 0, -7 }, // 'W'
00159     { 289, 6, 7, 6, 0, -7 }, // 'X'
00160     { 295, 8, 7, 8, 0, -7 }, // 'Y'
00161     { 302, 7, 7, 7, 0, -7 }, // 'Z'
00162     { 309, 3, 9, 4, 1, -7 }, // '['
00163     { 313, 4, 7, 4, 0, -7 }, // '\'
00164     { 317, 3, 9, 4, 0, -7 }, // ']'
00165     { 321, 4, 3, 4, 0, -7 }, // '^'
00166     { 323, 6, 1, 6, 0, 1 }, // '_'
00167     { 324, 3, 2, 4, 0, -8 }, // '`'
00168     { 325, 6, 5, 6, 0, -5 }, // 'a'
00169     { 329, 5, 7, 6, 1, -7 }, // 'b'
00170     { 334, 6, 5, 6, 0, -5 }, // 'c'
00171     { 338, 5, 7, 6, 0, -7 }, // 'd'
00172     { 343, 6, 5, 6, 0, -5 }, // 'e'
00173     { 347, 5, 7, 5, 0, -7 }, // 'f'
00174     { 352, 5, 7, 6, 0, -5 }, // 'g'
00175     { 357, 4, 7, 6, 1, -7 }, // 'h'
00176     { 361, 2, 7, 3, 1, -7 }, // 'i'
00177     { 363, 2, 9, 3, 0, -7 }, // 'j'
00178     { 366, 5, 7, 6, 1, -7 }, // 'k'
00179     { 371, 2, 7, 3, 1, -7 }, // 'l'
00180     { 373, 7, 5, 9, 1, -5 }, // 'm'
00181     { 378, 4, 5, 6, 1, -5 }, // 'n'
00182     { 381, 6, 5, 6, 0, -5 }, // 'o'
00183     { 385, 5, 7, 6, 1, -5 }, // 'p'
00184     { 390, 5, 7, 6, 0, -5 }, // 'q'
00185     { 395, 3, 5, 4, 1, -5 }, // 'r'
00186     { 397, 6, 5, 6, 0, -5 }, // 's'
00187     { 401, 4, 6, 4, 0, -6 }, // 't'
00188     { 404, 4, 5, 6, 1, -5 }, // 'u'
00189     { 407, 7, 5, 7, 0, -5 }, // 'v'
00190     { 412, 7, 5, 7, 0, -5 }, // 'w'
00191     { 417, 6, 5, 6, 0, -5 }, // 'x'
00192     { 421, 7, 7, 7, 0, -5 }, // 'y'
00193     { 428, 5, 5, 5, 0, -5 }, // 'z'
00194     { 432, 4, 9, 4, 0, -7 }, // '{'
00195     { 437, 2, 9, 4, 1, -7 }, // '|'
00196     { 440, 4, 9, 4, 0, -7 }, // '}'
00197 };
00198 const GFXfont Arimo_Regular_9 PROGMEM = {
00199 (uint8_t *)Arimo_Regular_9Bitmaps, (GFXglyph *)Arimo_Regular_9Glyphs, 0x20, 0x7E, 12};

```

## 2.6 custom\_fonts.h

```

00001 #include <custom_fonts/arimo9.h>
00002 #include <custom_fonts/arimo12.h>
00003 #include <custom_fonts/dialog9.h>
00004 #include <custom_fonts/dialog12.h>
00005 #include <custom_fonts/gothic9.h>
00006 #include <custom_fonts/gothic12.h>
00007 #include <custom_fonts/mono9.h>
00008 #include <custom_fonts/mono12.h>
00009 #include <custom_fonts/opensans9.h>
00010 #include <custom_fonts/opensans12.h>
00011 #include <custom_fonts/roboto9.h>
00012 #include <custom_fonts/roboto12.h>
00013 #include <custom_fonts/SansSerif9.h>
00014 #include <custom_fonts/SansSerif12.h>
00015 #include <custom_fonts/serif9.h>
00016 #include <custom_fonts/serif12.h>
00017
00018
00019
00020
00021 #define arimo9 &Arimo_Regular_9

```

```

00022 #define arimo12 &Arimo_Regular_12
00023 #define dialog9 &Dialog_plain_9
00024 #define dialog12 &Dialog_plain_12
00025 #define gothic9 &URW_Gothic_L_Book_9
00026 #define gothic12 &URW_Gothic_L_Book_12
00027 #define roboto9 &Roboto_Condensed_Light_9
00028 #define roboto12 &Roboto_Condensed_Light_12
00029 #define mono9 &Monospaced_plain_9
00030 #define mono12 &Monospaced_plain_12
00031 #define opensans9 &Open_Sans_Regular_9
00032 #define opensans12 &Open_Sans_Regular_12
00033 #define sansserif9 &SansSerif_plain_9
00034 #define sansserif12 &SansSerif_plain_12
00035 #define serif9 &Serif_plain_9
00036 #define serif12 &Serif_plain_12

```

## 2.7 dialog12.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Dialog_plain_12Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA,0xA2,0x80, // '!'
00008     0xAA,0xA0, // '" '
00009     0x12,0x0A,0x1F,0xC4,0x82,0x47,0xF0,0xA0,0x90, // '#'
00010     0x21,0xCA,0xA8,0xE0,0xE2,0xAA,0x70,0x82,0x00, // '$'
00011     0x61,0x12,0x42,0x48,0x4A,0x06,0xD8,0x14,0x84,0x90,0x92,0x21,0x80, // '%'
00012     0x30,0x24,0x10,0x0C,0x05,0x14,0x4A,0x19,0x88,0x7B,0x00, // '&'
00013     0xA8, // '" '
00014     0x64,0x48,0x88,0x88,0x44,0x60, // '('
00015     0xC4,0x42,0x22,0x22,0x44,0xC0, // ')'
00016     0x22,0xA7,0x1C,0xA8,0x80, // '*'
00017     0x10,0x10,0x10,0xFE,0x10,0x10,0x10, // '+'
00018     0xA8, // ', '
00019     0xE0, // '- '
00020     0xA0, // '. '
00021     0x10,0x88,0x42,0x21,0x08,0x84,0x00, // '/'
00022     0x78,0x92,0x14,0x28,0x50,0xA1,0x24,0x78, // '0'
00023     0xE0,0x82,0x08,0x20,0x82,0x08,0xF8, // '1'
00024     0x79,0x18,0x10,0x20,0x82,0x08,0x20,0xFC, // '2'
00025     0x79,0x08,0x10,0x23,0x80,0x81,0x42,0x78, // '3'
00026     0x18,0x30,0xA2,0x44,0x91,0x3F,0x04,0x08, // '4'
00027     0xF9,0x02,0x07,0xC0,0xC0,0x81,0x46,0x78, // '5'
00028     0x38,0x8A,0x05,0xCC,0xD0,0xA1,0x26,0x78, // '6'
00029     0xFC,0x08,0x20,0x41,0x02,0x08,0x10,0x40, // '7'
00030     0x79,0x0A,0x14,0x27,0x90,0xA1,0x42,0x78, // '8'
00031     0x79,0x92,0x14,0x2C,0xCE,0x81,0x44,0x70, // '9'
00032     0xA0,0xA0, // ':'
00033     0xA0,0xA8, // ';'
00034     0x03,0x0F,0x38,0x1C,0x01,0xE0,0x18, // '<'
00035     0xFF,0x00,0x3F,0xC0, // '='
00036     0xC0,0x3C,0x01,0xC0,0xE7,0x86,0x00, // '>'
00037     0x72,0x20,0x84,0x20,0x80,0x08,0x20, // '?'
00038     0x1F,0x02,0x0C,0x40,0x48,0xF2,0x91,0x29,0x12,0x91,0x48,0xF8,0x40,0x02,0x08,0x1F,0x00, // '@'
00039     0x18,0x0C,0x09,0x04,0x82,0x42,0x11,0xF8,0x84,0x81,0x00, // 'A'
00040     0xF9,0x0A,0x14,0x2F,0x90,0xA1,0x42,0xF8, // 'B'
00041     0x38,0x8A,0x04,0x08,0x10,0x20,0x22,0x38, // 'C'
00042     0xF8,0x84,0x82,0x82,0x82,0x82,0x84,0xF8, // 'D'
00043     0xFD,0x02,0x04,0x0F,0xD0,0x20,0x40,0xFC, // 'E'
00044     0xFA,0x08,0x20,0xFA,0x08,0x20,0x80, // 'F'
00045     0x3C,0x42,0x80,0x80,0x8E,0x82,0x82,0x42,0x3C, // 'G'
00046     0x82,0x82,0x82,0x82,0xFE,0x82,0x82,0x82, // 'H'
00047     0xAA,0xAA,0x80, // 'I'
00048     0x22,0x22,0x22,0x22,0x22,0xC0, // 'J'
00049     0x84,0x88,0x90,0xA0,0xC0,0xA0,0x90,0x88,0x84, // 'K'
00050     0x82,0x08,0x20,0x82,0x08,0x20,0xF8, // 'L'
00051     0x81,0x61,0xB0,0xD4,0xAA,0x54,0xCA,0x65,0x02,0x81,0x00, // 'M'
00052     0xC2,0xC2,0xA2,0xA2,0x92,0x8A,0x8A,0x86,0x86, // 'N'
00053     0x38,0x44,0x82,0x82,0x82,0x82,0x82,0x44,0x38, // 'O'
00054     0xF9,0x0A,0x14,0x2F,0x90,0x20,0x40,0x80, // 'P'
00055     0x38,0x44,0x82,0x82,0x82,0x82,0x44,0x38,0x08,0x04, // 'Q'
00056     0xF8,0x84,0x84,0x84,0xF8,0x88,0x84,0x84,0x82, // 'R'
00057     0x79,0x0A,0x04,0x07,0x80,0x81,0x42,0x78, // 'S'
00058     0xFE,0x10,0x10,0x10,0x10,0x10,0x10,0x10, // 'T'
00059     0x82,0x82,0x82,0x82,0x82,0x82,0x82,0xC6,0x7C, // 'U'
00060     0x40,0x88,0x10,0x84,0x10,0x82,0x10,0x24,0x04,0x80,0x60,0x0C,0x00, // 'V'
00061     0x84,0x24,0x44,0x44,0x44,0xA4,0x2A,0x82,0xA8,0x2A,0x81,0x10,0x11,0x00, // 'W'
00062     0xC6,0x44,0x28,0x28,0x10,0x28,0x28,0x44,0x82, // 'X'
00063     0x82,0x44,0x44,0x28,0x28,0x10,0x10,0x10,0x10, // 'Y'
00064     0xFE,0x02,0x04,0x08,0x10,0x20,0x40,0x80,0xFE, // 'Z'
00065     0xD2,0x49,0x24,0x93,0x00, // '['

```

```

00066     0x84, 0x10, 0x84, 0x10, 0x84, 0x10, 0x80, // '\
00067     0xC9, 0x24, 0x92, 0x4B, 0x00, // ']'
00068     0x18, 0x12, 0x10, 0x80, // '^'
00069     0xFC, // '._'
00070     0x42, // '._'
00071     0x79, 0x08, 0x13, 0xE8, 0x51, 0x9D, 0x00, // 'a'
00072     0x81, 0x02, 0x07, 0xCC, 0xD0, 0xA1, 0x42, 0xCD, 0xF0, // 'b'
00073     0x73, 0x28, 0x20, 0x83, 0x27, 0x00, // 'c'
00074     0x04, 0x08, 0x13, 0xEC, 0xD0, 0xA1, 0x42, 0xCC, 0xF8, // 'd'
00075     0x79, 0x9A, 0x17, 0xE8, 0x18, 0x9E, 0x00, // 'e'
00076     0x32, 0x11, 0xE4, 0x21, 0x08, 0x42, 0x00, // 'f'
00077     0x7D, 0x9A, 0x14, 0x28, 0x59, 0x9F, 0x02, 0x4C, 0x70, // 'g'
00078     0x81, 0x02, 0x05, 0xCC, 0x50, 0xA1, 0x42, 0x85, 0x08, // 'h'
00079     0x8A, 0xAA, 0x80, // 'i'
00080     0x41, 0x24, 0x92, 0x49, 0x60, // 'j'
00081     0x81, 0x02, 0x04, 0x49, 0x14, 0x30, 0x50, 0x91, 0x10, // 'k'
00082     0xAA, 0xAA, 0xA0, // 'l'
00083     0xF7, 0x22, 0x28, 0x8A, 0x22, 0x88, 0xA2, 0x28, 0x88, // 'm'
00084     0xB9, 0x8A, 0x14, 0x28, 0x50, 0xA1, 0x00, // 'n'
00085     0x79, 0x9A, 0x14, 0x28, 0x59, 0x9E, 0x00, // 'o'
00086     0xF9, 0x9A, 0x14, 0x28, 0x59, 0xBE, 0x40, 0x81, 0x00, // 'p'
00087     0x7D, 0x9A, 0x14, 0x28, 0x59, 0x9F, 0x02, 0x04, 0x08, // 'q'
00088     0xB6, 0x21, 0x08, 0x42, 0x00, // 'r'
00089     0x72, 0x28, 0x1C, 0x0A, 0x27, 0x00, // 's'
00090     0x42, 0x3C, 0x84, 0x21, 0x08, 0x70, // 't'
00091     0x85, 0x0A, 0x14, 0x28, 0x51, 0x9D, 0x00, // 'u'
00092     0x85, 0x09, 0x22, 0x44, 0x86, 0x0C, 0x00, // 'v'
00093     0x88, 0xA2, 0x25, 0x51, 0x54, 0x55, 0x08, 0x82, 0x20, // 'w'
00094     0x84, 0x91, 0x21, 0x84, 0x89, 0x21, 0x00, // 'x'
00095     0x85, 0x09, 0x22, 0x42, 0x86, 0x04, 0x08, 0x21, 0x80, // 'y'
00096     0xF8, 0x21, 0x08, 0x42, 0x0F, 0x80, // 'z'
00097     0x38, 0x82, 0x08, 0x23, 0x02, 0x08, 0x20, 0x83, 0x80, // '{'
00098     0xAA, 0xAA, 0xAA, // '|'
00099     0xE0, 0x82, 0x08, 0x20, 0x62, 0x08, 0x20, 0x8E, 0x00 // '}'
00100 };
00101 const GFXglyph Dialog_plain_l2Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103 { 0, 2, 1, 5, 0, -1 }, // ' '
00104 { 1, 2, 9, 6, 2, -9 }, // '!'
00105 { 4, 4, 3, 6, 1, -9 }, // '"'
00106 { 6, 9, 8, 11, 1, -8 }, // '#'
00107 { 15, 6, 11, 9, 2, -9 }, // '$'
00108 { 24, 11, 9, 12, 0, -9 }, // '%'
00109 { 37, 9, 9, 11, 1, -9 }, // '&'
00110 { 48, 2, 3, 4, 1, -9 }, // "'"
00111 { 49, 4, 11, 6, 1, -10 }, // '('
00112 { 55, 4, 11, 6, 1, -10 }, // ')'
00113 { 61, 6, 6, 7, 1, -9 }, // '*'
00114 { 66, 8, 7, 11, 1, -7 }, // '+'
00115 { 73, 2, 3, 5, 1, -2 }, // ','
00116 { 74, 4, 1, 5, 1, -4 }, // '-'
00117 { 75, 2, 2, 5, 1, -2 }, // '.'
00118 { 76, 5, 10, 5, 0, -9 }, // '/'
00119 { 83, 7, 9, 9, 1, -9 }, // '0'
00120 { 91, 6, 9, 9, 1, -9 }, // '1'
00121 { 98, 7, 9, 9, 1, -9 }, // '2'
00122 { 106, 7, 9, 9, 1, -9 }, // '3'
00123 { 114, 7, 9, 9, 1, -9 }, // '4'
00124 { 122, 7, 9, 9, 1, -9 }, // '5'
00125 { 130, 7, 9, 9, 1, -9 }, // '6'
00126 { 138, 7, 9, 9, 1, -9 }, // '7'
00127 { 146, 7, 9, 9, 1, -9 }, // '8'
00128 { 154, 7, 9, 9, 1, -9 }, // '9'
00129 { 162, 2, 6, 5, 1, -6 }, // ':'
00130 { 164, 2, 7, 5, 1, -6 }, // ';'
00131 { 166, 9, 6, 11, 1, -7 }, // '<'
00132 { 173, 9, 3, 11, 1, -5 }, // '='
00133 { 177, 9, 6, 11, 1, -7 }, // '>'
00134 { 184, 6, 9, 7, 0, -9 }, // '?'
00135 { 191, 12, 11, 14, 1, -9 }, // '@'
00136 { 208, 9, 9, 9, 0, -9 }, // 'A'
00137 { 219, 7, 9, 9, 1, -9 }, // 'B'
00138 { 227, 7, 9, 9, 1, -9 }, // 'C'
00139 { 235, 8, 9, 10, 1, -9 }, // 'D'
00140 { 244, 7, 9, 9, 1, -9 }, // 'E'
00141 { 252, 6, 9, 8, 1, -9 }, // 'F'
00142 { 259, 8, 9, 10, 1, -9 }, // 'G'
00143 { 268, 8, 9, 10, 1, -9 }, // 'H'
00144 { 277, 2, 9, 4, 1, -9 }, // 'I'
00145 { 280, 4, 11, 4, -1, -9 }, // 'J'
00146 { 286, 8, 9, 8, 1, -9 }, // 'K'
00147 { 295, 6, 9, 7, 1, -9 }, // 'L'
00148 { 302, 9, 9, 11, 1, -9 }, // 'M'
00149 { 313, 8, 9, 10, 1, -9 }, // 'N'
00150 { 322, 8, 9, 10, 1, -9 }, // 'O'
00151 { 331, 7, 9, 9, 1, -9 }, // 'P'
00152 { 339, 8, 11, 10, 1, -9 }, // 'Q'

```

```

00153 { 350, 8, 9, 9, 1, -9 }, // 'R'
00154 { 359, 7, 9, 9, 1, -9 }, // 'S'
00155 { 367, 8, 9, 8, 0, -9 }, // 'T'
00156 { 376, 8, 9, 10, 1, -9 }, // 'U'
00157 { 385, 11, 9, 9, -1, -9 }, // 'V'
00158 { 398, 12, 9, 12, 0, -9 }, // 'W'
00159 { 412, 8, 9, 8, 0, -9 }, // 'X'
00160 { 421, 8, 9, 8, 0, -9 }, // 'Y'
00161 { 430, 8, 9, 10, 1, -9 }, // 'Z'
00162 { 439, 3, 11, 6, 2, -9 }, // '['
00163 { 444, 5, 10, 5, 0, -9 }, // '\'
00164 { 451, 3, 11, 6, 1, -9 }, // ']'
00165 { 456, 9, 3, 11, 1, -9 }, // '^'
00166 { 460, 7, 1, 7, 0, 2 }, // '_'
00167 { 461, 4, 2, 7, 1, -10 }, // '`'
00168 { 462, 7, 7, 9, 1, -7 }, // 'a'
00169 { 469, 7, 10, 9, 1, -10 }, // 'b'
00170 { 478, 6, 7, 8, 1, -7 }, // 'c'
00171 { 484, 7, 10, 9, 1, -10 }, // 'd'
00172 { 493, 7, 7, 9, 1, -7 }, // 'e'
00173 { 500, 5, 10, 5, 0, -10 }, // 'f'
00174 { 507, 7, 10, 9, 1, -7 }, // 'g'
00175 { 516, 7, 10, 9, 1, -10 }, // 'h'
00176 { 525, 2, 9, 4, 1, -9 }, // 'i'
00177 { 528, 3, 12, 4, 0, -9 }, // 'j'
00178 { 533, 7, 10, 8, 1, -10 }, // 'k'
00179 { 542, 2, 10, 4, 1, -10 }, // 'l'
00180 { 545, 10, 7, 12, 1, -7 }, // 'm'
00181 { 554, 7, 7, 9, 1, -7 }, // 'n'
00182 { 561, 7, 7, 9, 1, -7 }, // 'o'
00183 { 568, 7, 10, 9, 1, -7 }, // 'p'
00184 { 577, 7, 10, 9, 1, -7 }, // 'q'
00185 { 586, 5, 7, 6, 1, -7 }, // 'r'
00186 { 591, 6, 7, 8, 1, -7 }, // 's'
00187 { 597, 5, 9, 6, 0, -9 }, // 't'
00188 { 603, 7, 7, 9, 1, -7 }, // 'u'
00189 { 610, 7, 7, 7, 0, -7 }, // 'v'
00190 { 617, 10, 7, 10, 0, -7 }, // 'w'
00191 { 626, 7, 7, 7, 0, -7 }, // 'x'
00192 { 633, 7, 10, 7, 0, -7 }, // 'y'
00193 { 642, 6, 7, 6, 0, -7 }, // 'z'
00194 { 648, 6, 11, 9, 2, -9 }, // '{'
00195 { 657, 2, 12, 5, 2, -9 }, // '|'
00196 { 660, 6, 11, 9, 1, -9 }, // '}'
00197 };
00198 const GFXfont Dialog_plain_12 PROGMEM = {
00199 (uint8_t *)Dialog_plain_12Bitmaps, (GFXglyph *)Dialog_plain_12Glyphs, 0x20, 0x7E, 15};

```

## 2.8 dialog9.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Dialog_plain_9Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0x88, // '!'
00008     0xAA, // '"'
00009     0x28, 0x53, 0xF1, 0x4F, 0xCA, 0x14, 0x00, // '#'
00010     0x21, 0xEA, 0x38, 0x38, 0xAF, 0x08, // '$'
00011     0xE4, 0x52, 0x2A, 0x1F, 0xE1, 0x51, 0x28, 0x9C, // '%'
00012     0x30, 0x91, 0x03, 0x29, 0x53, 0x1B, 0x00, // '&'
00013     0xA0, // "'"
00014     0x52, 0x49, 0x22, // '('
00015     0x91, 0x24, 0xA4, // ')'
00016     0xA9, 0xC7, 0x2A, // '*'
00017     0x20, 0x8F, 0x88, 0x20, // '+'
00018     0xA0, // ','
00019     0xC0, // '-'
00020     0x80, // '.'
00021     0x22, 0x44, 0x48, 0x80, // '/'
00022     0x64, 0xA5, 0x29, 0x49, 0x80, // '0'
00023     0xC4, 0x44, 0x44, 0xE0, // '1'
00024     0x64, 0x84, 0x44, 0x43, 0xC0, // '2'
00025     0x64, 0x84, 0xC1, 0x0B, 0x80, // '3'
00026     0x10, 0xC5, 0x14, 0x93, 0xE1, 0x00, // '4'
00027     0xF4, 0x21, 0xC1, 0x0B, 0x80, // '5'
00028     0x76, 0x21, 0xC9, 0x49, 0x80, // '6'
00029     0xF0, 0x88, 0x42, 0x21, 0x00, // '7'
00030     0x64, 0xA4, 0xC9, 0x49, 0x80, // '8'
00031     0x64, 0xA5, 0xE1, 0x1B, 0x80, // '9'
00032     0x80, 0x80, // ':'
00033     0x80, 0xA0, // ';'

```

```

00034     0x04,0x73,0x01,0xC0,0x40, // '<'
00035     0xFC,0x03,0xF0, // '='
00036     0x80,0xE0,0x33,0x88,0x00, // '>'
00037     0xF0,0x88,0x84,0x01,0x00, // '?'
00038     0x3C,0x21,0x27,0x54,0xAA,0x54,0xF1,0x10,0x70, // '@'
00039     0x30,0x61,0x22,0x47,0x90,0xA1,0x00, // 'A'
00040     0xF2,0x28,0xBC,0x8A,0x2F,0x00, // 'B'
00041     0x73,0x28,0x20,0x83,0x07,0x80, // 'C'
00042     0xF2,0x68,0xA2,0x8A,0x6F,0x00, // 'D'
00043     0xF4,0x21,0xE8,0x43,0xC0, // 'E'
00044     0xF4,0x21,0xE8,0x42,0x00, // 'F'
00045     0x73,0x28,0x26,0x8B,0x27,0x00, // 'G'
00046     0x8A,0x28,0xBE,0x8A,0x28,0x80, // 'H'
00047     0xAA,0xA8, // 'I'
00048     0x49,0x24,0x92,0x80, // 'J'
00049     0x8A,0x4A,0x30,0xA2,0x48,0x80, // 'K'
00050     0x84,0x21,0x08,0x43,0xC0, // 'L'
00051     0x85,0x9B,0x35,0xAB,0x50,0xA1,0x00, // 'M'
00052     0x8B,0x2C,0xAA,0x9A,0x68,0x80, // 'N'
00053     0x73,0x68,0xA2,0x8B,0x67,0x00, // 'O'
00054     0xE4,0xA5,0xC8,0x42,0x00, // 'P'
00055     0x73,0x68,0xA2,0x8B,0x67,0x04, // 'Q'
00056     0xF2,0x49,0x38,0xA2,0x48,0x80, // 'R'
00057     0x72,0x28,0x1C,0x0A,0x27,0x00, // 'S'
00058     0xF8,0x82,0x08,0x20,0x82,0x00, // 'T'
00059     0x8A,0x28,0xA2,0x8A,0x27,0x00, // 'U'
00060     0x85,0x09,0x22,0x44,0x86,0x0C,0x00, // 'V'
00061     0x92,0x92,0x54,0x54,0x6C,0x28,0x28, // 'W'
00062     0xCC,0x90,0xC1,0x83,0x09,0x23,0x00, // 'X'
00063     0x89,0x45,0x08,0x20,0x82,0x00, // 'Y'
00064     0xF8,0x21,0x08,0x42,0x0F,0x80, // 'Z'
00065     0xD2,0x49,0x26, // '['
00066     0x88,0x44,0x42,0x20, // '\ '
00067     0xC9,0x24,0x96, // ']'
00068     0x30,0x90, // '^'
00069     0xF8, // '_'
00070     0x88, // '`'
00071     0x70,0xBD,0x2F,0x00, // 'a'
00072     0x84,0x21,0xC9,0x4A,0x5C, // 'b'
00073     0x74,0x21,0x07,0x00, // 'c'
00074     0x10,0x84,0xE9,0x4A,0x4E, // 'd'
00075     0x64,0xBD,0x07,0x00, // 'e'
00076     0x72,0x11,0xC4,0x21,0x08, // 'f'
00077     0x74,0xA5,0x27,0x09,0x80, // 'g'
00078     0x84,0x21,0xE9,0x4A,0x52, // 'h'
00079     0x8A,0xA8, // 'i'
00080     0x41,0x24,0x92,0xC0, // 'j'
00081     0x84,0x21,0x2A,0x62,0x92, // 'k'
00082     0xAA,0xAA, // 'l'
00083     0xFE,0x92,0x92,0x92,0x92, // 'm'
00084     0xF4,0xA5,0x29,0x00, // 'n'
00085     0x64,0xA5,0x26,0x00, // 'o'
00086     0xE4,0xA5,0x2E,0x42,0x00, // 'p'
00087     0x74,0xA5,0x27,0x08,0x40, // 'q'
00088     0xE8,0x88,0x80, // 'r'
00089     0xE8,0x62,0xE0, // 's'
00090     0x47,0x90,0x84,0x38, // 't'
00091     0x94,0xA5,0x2F,0x00, // 'u'
00092     0x8A,0x25,0x14,0x20, // 'v'
00093     0x92,0xAA,0xAA,0x44,0x44, // 'w'
00094     0x89,0x42,0x14,0x88, // 'x'
00095     0x89,0x11,0x42,0x82,0x04,0x30,0x00, // 'y'
00096     0xF0,0x88,0x8F,0x00, // 'z'
00097     0x64,0x48,0x44,0x46, // '{'
00098     0xAA,0xAA,0x80, // '|'
00099     0xC4,0x42,0x4A,0x4C // '}'
00100 };
00101 const GFXglyph Dialog_plain_9Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103 { 0, 2, 1, 4, 0, -1 }, // ' '
00104 { 1, 2, 7, 4, 1, -7 }, // '!'
00105 { 3, 4, 2, 5, 1, -7 }, // '"'
00106 { 4, 7, 7, 9, 1, -7 }, // '#'
00107 { 11, 6, 8, 7, 0, -7 }, // '$'
00108 { 17, 9, 7, 10, 0, -7 }, // '%'
00109 { 25, 7, 7, 9, 1, -7 }, // '&'
00110 { 32, 2, 2, 3, 1, -7 }, // "'"
00111 { 33, 3, 8, 5, 1, -8 }, // '('
00112 { 36, 3, 8, 5, 1, -8 }, // ')'
00113 { 39, 6, 4, 6, 0, -7 }, // '*'
00114 { 42, 6, 5, 9, 1, -5 }, // '+'
00115 { 46, 2, 2, 4, 1, -1 }, // ','
00116 { 47, 3, 1, 4, 1, -3 }, // '-'
00117 { 48, 2, 1, 4, 1, -1 }, // '.'
00118 { 49, 4, 7, 4, 0, -7 }, // '/'
00119 { 53, 5, 7, 7, 1, -7 }, // '0'
00120 { 58, 4, 7, 7, 2, -7 }, // '1'

```

```

00121 { 62, 5, 7, 7, 1, -7 }, // '2'
00122 { 67, 5, 7, 7, 1, -7 }, // '3'
00123 { 72, 6, 7, 7, 1, -7 }, // '4'
00124 { 78, 5, 7, 7, 1, -7 }, // '5'
00125 { 83, 5, 7, 7, 1, -7 }, // '6'
00126 { 88, 5, 7, 7, 1, -7 }, // '7'
00127 { 93, 5, 7, 7, 1, -7 }, // '8'
00128 { 98, 5, 7, 7, 1, -7 }, // '9'
00129 { 103, 2, 5, 4, 1, -5 }, // ':'
00130 { 105, 2, 6, 4, 1, -5 }, // ';'
00131 { 107, 7, 5, 9, 1, -5 }, // '<'
00132 { 112, 7, 3, 9, 1, -4 }, // '='
00133 { 115, 7, 5, 9, 1, -5 }, // '>'
00134 { 120, 5, 7, 6, 1, -7 }, // '?'
00135 { 125, 9, 8, 11, 1, -7 }, // '@'
00136 { 134, 7, 7, 7, 0, -7 }, // 'A'
00137 { 141, 6, 7, 8, 1, -7 }, // 'B'
00138 { 147, 6, 7, 8, 1, -7 }, // 'C'
00139 { 153, 6, 7, 8, 1, -7 }, // 'D'
00140 { 159, 5, 7, 7, 1, -7 }, // 'E'
00141 { 164, 5, 7, 7, 1, -7 }, // 'F'
00142 { 169, 6, 7, 8, 1, -7 }, // 'G'
00143 { 175, 6, 7, 8, 1, -7 }, // 'H'
00144 { 181, 2, 7, 4, 1, -7 }, // 'I'
00145 { 183, 3, 9, 4, 0, -7 }, // 'J'
00146 { 187, 6, 7, 7, 1, -7 }, // 'K'
00147 { 193, 5, 7, 6, 1, -7 }, // 'L'
00148 { 198, 7, 7, 9, 1, -7 }, // 'M'
00149 { 205, 6, 7, 8, 1, -7 }, // 'N'
00150 { 211, 6, 7, 8, 1, -7 }, // 'O'
00151 { 217, 5, 7, 7, 1, -7 }, // 'P'
00152 { 222, 6, 8, 8, 1, -7 }, // 'Q'
00153 { 228, 6, 7, 7, 1, -7 }, // 'R'
00154 { 234, 6, 7, 8, 1, -7 }, // 'S'
00155 { 240, 6, 7, 6, 0, -7 }, // 'T'
00156 { 246, 6, 7, 8, 1, -7 }, // 'U'
00157 { 252, 7, 7, 7, 0, -7 }, // 'V'
00158 { 259, 8, 7, 8, 0, -7 }, // 'W'
00159 { 266, 7, 7, 7, 0, -7 }, // 'X'
00160 { 273, 6, 7, 6, 0, -7 }, // 'Y'
00161 { 279, 6, 7, 6, 0, -7 }, // 'Z'
00162 { 285, 3, 8, 5, 1, -7 }, // '['
00163 { 288, 4, 7, 4, 0, -7 }, // '\'
00164 { 292, 3, 8, 5, 1, -7 }, // ']'
00165 { 295, 7, 2, 9, 1, -7 }, // '^'
00166 { 297, 6, 1, 6, 0, 1 }, // '_'
00167 { 298, 3, 2, 6, 1, -8 }, // '`'
00168 { 299, 5, 5, 7, 1, -5 }, // 'a'
00169 { 303, 5, 8, 7, 1, -8 }, // 'b'
00170 { 308, 5, 5, 7, 1, -5 }, // 'c'
00171 { 312, 5, 8, 7, 1, -8 }, // 'd'
00172 { 317, 5, 5, 7, 1, -5 }, // 'e'
00173 { 321, 5, 8, 4, 0, -8 }, // 'f'
00174 { 326, 5, 7, 7, 1, -5 }, // 'g'
00175 { 331, 5, 8, 7, 1, -8 }, // 'h'
00176 { 336, 2, 7, 4, 1, -7 }, // 'i'
00177 { 338, 3, 9, 4, 0, -7 }, // 'j'
00178 { 342, 5, 8, 6, 1, -8 }, // 'k'
00179 { 347, 2, 8, 4, 1, -8 }, // 'l'
00180 { 349, 8, 5, 10, 1, -5 }, // 'm'
00181 { 354, 5, 5, 7, 1, -5 }, // 'n'
00182 { 358, 5, 5, 7, 1, -5 }, // 'o'
00183 { 362, 5, 7, 7, 1, -5 }, // 'p'
00184 { 367, 5, 7, 7, 1, -5 }, // 'q'
00185 { 372, 4, 5, 5, 1, -5 }, // 'r'
00186 { 375, 4, 5, 6, 1, -5 }, // 's'
00187 { 378, 5, 6, 5, 0, -6 }, // 't'
00188 { 382, 5, 5, 7, 1, -5 }, // 'u'
00189 { 386, 6, 5, 6, 0, -5 }, // 'v'
00190 { 390, 8, 5, 8, 0, -5 }, // 'w'
00191 { 395, 6, 5, 6, 0, -5 }, // 'x'
00192 { 399, 7, 7, 6, 0, -5 }, // 'y'
00193 { 406, 5, 5, 7, 1, -5 }, // 'z'
00194 { 410, 4, 8, 6, 1, -7 }, // '{'
00195 { 414, 2, 9, 4, 1, -7 }, // '|'
00196 { 417, 4, 8, 6, 1, -7 } // '}'
00197 };
00198 const GFXfont Dialog_plain_9 PROGMEM = {
00199 (uint8_t *)Dialog_plain_9Bitmaps, (GFXglyph *)Dialog_plain_9Glyphs, 0x20, 0x7E, 12};

```

## 2.9 gothic12.h

```
00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
```

```

00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t URW_Gothic_L_Book_12Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA,0xA8,0x80, // '!'
00008     0xDB,0x00, // '"'
00009     0x24,0x24,0xFE,0x48,0x48,0xFC,0x48,0x48,0x50, // '#'
00010     0x21,0xC8,0xA0,0xC0,0xC0,0xA2,0x89,0xC2,0x00, // '$'
00011     0x62,0x25,0x09,0x41,0xA0,0x08,0x04,0xC1,0x48,0x92,0x23,0x00, // '%'
00012     0x38,0x24,0x12,0x07,0x27,0x24,0x72,0x11,0x96,0x79,0x00, // '&'
00013     0xA8, // '''
00014     0x32,0x11,0x08,0x42,0x10,0x41,0x04, // '('
00015     0xC1,0x08,0x21,0x08,0x46,0x22,0x20, // ')'
00016     0x04,0xEC, // '*'
00017     0x10,0x10,0xFE,0x10,0x10,0x10, // '+'
00018     0x4A,0x00, // ','
00019     0xF0, // '-'
00020     0x80, // '.'
00021     0x08,0x41,0x04,0x20,0x82,0x10,0x41,0x08,0x00, // '/'
00022     0x79,0x9A,0x14,0x28,0x50,0xA1,0x66,0x78, // '0'
00023     0xE2,0x22,0x22,0x22,0x20, // '1'
00024     0x79,0x9A,0x10,0x20,0x82,0x08,0x20,0xFC, // '2'
00025     0x70,0x90,0x21,0x80,0xC0,0xA1,0x66,0x78, // '3'
00026     0x0C,0x0C,0x14,0x24,0x24,0x44,0xFE,0x04,0x04, // '4'
00027     0x78,0x81,0xE6,0x40,0x40,0xA1,0x64,0x78, // '5'
00028     0x10,0x40,0xE2,0x68,0x50,0xA1,0x66,0x78, // '6'
00029     0xF8,0x21,0x04,0x20,0x82,0x10,0x40, // '7'
00030     0x71,0x12,0x23,0x8C,0xD0,0xA1,0x66,0x78, // '8'
00031     0x79,0x9A,0x14,0x2C,0xCF,0x04,0x08,0x20, // '9'
00032     0x80,0x20, // ':'
00033     0x40,0x05,0x20, // ';'
00034     0x04,0x31,0x86,0x03,0x01,0x80,0x00, // '<'
00035     0xFC,0x03,0xF0, // '='
00036     0x80,0xC0,0x60,0x63,0x18,0x00,0x00, // '>'
00037     0x72,0x68,0x82,0x10,0xC2,0x00,0x20, // '?'
00038     0x3E,0x1B,0xCD,0x2A,0x92,0xA4,0xA9,0x4F,0xF1,0x84,0x3E,0x00, // '@'
00039     0x08,0x05,0x01,0x40,0x90,0x22,0x0F,0x84,0x11,0x04,0x80,0x80, // 'A'
00040     0xF1,0x12,0x24,0x4F,0x11,0xA1,0x42,0xF8, // 'B'
00041     0x3E,0x18,0x4C,0x0A,0x00,0x80,0x20,0x0C,0x09,0x84,0x3E,0x00, // 'C'
00042     0xF8,0x43,0x20,0xD0,0x28,0x14,0x0A,0x0D,0x0C,0xFC,0x00, // 'D'
00043     0xFA,0x08,0x20,0xFA,0x08,0x20,0xF8, // 'E'
00044     0xFA,0x08,0x20,0xF2,0x08,0x20,0x80, // 'F'
00045     0x3E,0x0C,0x33,0x00,0x40,0x08,0xF9,0x01,0x10,0x23,0x08,0x1E,0x00, // 'G'
00046     0x85,0x0A,0x14,0x2F,0xD0,0xA1,0x42,0x84, // 'H'
00047     0xAA,0xAA,0x80, // 'I'
00048     0x08,0x20,0x82,0x08,0x20,0xA2,0x70, // 'J'
00049     0x85,0x12,0x45,0x0A,0x1E,0x24,0x44,0x84, // 'K'
00050     0x82,0x08,0x20,0x82,0x08,0x20,0xF8, // 'L'
00051     0xC1,0xB0,0x6C,0x2A,0x8A,0xA2,0xA5,0x29,0x4A,0x62,0x88,0x80, // 'M'
00052     0xC2,0xC2,0xA2,0xB2,0x92,0x8A,0x8A,0x86,0x82, // 'N'
00053     0x3E,0x18,0xCC,0x1A,0x02,0x80,0xA0,0x2C,0x19,0x8C,0x3E,0x00, // 'O'
00054     0xF9,0x0A,0x14,0x2F,0x90,0x20,0x40,0x80, // 'P'
00055     0x1E,0x0C,0x33,0x03,0x40,0x28,0x05,0xF8,0x91,0xA3,0x0C,0x1E,0xC0, // 'Q'
00056     0xF9,0x1A,0x14,0x28,0xD6,0x24,0x44,0x88, // 'R'
00057     0x72,0x28,0x20,0x70,0x28,0xA2,0x70, // 'S'
00058     0xF8,0x82,0x08,0x20,0x82,0x08,0x20, // 'T'
00059     0x85,0x0A,0x14,0x28,0x50,0xA1,0x66,0x78, // 'U'
00060     0x81,0x20,0x90,0x88,0x42,0x41,0x20,0x50,0x30,0x08,0x00, // 'V'
00061     0x84,0x28,0x62,0x46,0x24,0xA4,0x4A,0x42,0x94,0x31,0x43,0x18,0x10,0x80, // 'W'
00062     0x42,0x44,0x28,0x18,0x10,0x28,0x24,0x44,0x82, // 'X'
00063     0x82,0x44,0x44,0x28,0x28,0x10,0x10,0x10,0x10, // 'Y'
00064     0xFC,0x10,0x60,0x82,0x04,0x10,0x40,0xFC, // 'Z'
00065     0x68,0x88,0x88,0x88,0x84,0x20, // '['
00066     0x82,0x04,0x10,0x20,0x81,0x04,0x10, // '\'
00067     0xC2,0x22,0x22,0x22,0x24,0x80, // ']'
00068     0x10,0x28,0x28,0x24,0x44,0x44,0x82, // '^'
00069     0xFC, // '_'
00070     0x84, // '`'
00071     0x3A,0x46,0x82,0x82,0x46,0x3A, // 'a'
00072     0x80,0x80,0xB8,0xC4,0x82,0x82,0x82,0xC4,0xB8, // 'b'
00073     0x38,0x44,0x80,0x80,0x80,0x44,0x38, // 'c'
00074     0x02,0x02,0x3A,0x46,0x82,0x82,0x82,0x46,0x3A, // 'd'
00075     0x38,0x44,0x82,0xFE,0x80,0x44,0x38, // 'e'
00076     0x32,0x3C,0x84,0x21,0x08,0x40, // 'f'
00077     0x3A,0x46,0x82,0x82,0x46,0x3A,0xC6,0x3C, // 'g'
00078     0x81,0x02,0xE6,0x68,0x50,0xA1,0x42,0x84, // 'h'
00079     0x8A,0xAA,0x80, // 'i'
00080     0x41,0x24,0x92,0x4A,0x00, // 'j'
00081     0x82,0x08,0xA4,0xA2,0x8E,0x24,0x88, // 'k'
00082     0xAA,0xAA,0x80, // 'l'
00083     0xF7,0x22,0x28,0x8A,0x22,0x88,0xA2,0x28,0x88, // 'm'
00084     0xB9,0x9A,0x14,0x28,0x50,0xA1,0x00, // 'n'
00085     0x38,0x44,0x82,0x82,0x82,0x44,0x38, // 'o'
00086     0xB8,0xC4,0x82,0x82,0xC4,0xB8,0x80,0x80, // 'p'
00087     0x3A,0x46,0x82,0x82,0x82,0x46,0x3A,0x02,0x02, // 'q'
00088     0xE8,0x88,0x88,0x80, // 'r'

```

```

00089      0x64,0xA0,0xC1,0x4B,0x80, // 's'
00090      0x44,0xE4,0x44,0x44,0x40, // 't'
00091      0x85,0x0A,0x14,0x28,0x59,0x9D,0x00, // 'u'
00092      0x84,0x44,0x44,0x48,0x28,0x30,0x10, // 'v'
00093      0x80,0x51,0x91,0x32,0x26,0x43,0x50,0x66,0x0C,0xC0, // 'w'
00094      0x88,0x90,0xC1,0x03,0x09,0x22,0x00, // 'x'
00095      0x84,0x89,0x22,0x43,0x06,0x04,0x10,0x20, // 'y'
00096      0xF8,0x42,0x08,0x42,0x0F,0x80, // 'z'
00097      0x64,0x44,0x48,0x44,0x44,0x60, // '{'
00098      0xAA,0xAA,0x80, // '|'
00099      0xC4,0x44,0x42,0x44,0x44,0xC0 // '}'
00100  };
00101  const GFXglyph URW_Gothic_L_Book_12Glyphs[] PROGMEM = {
00102  // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103      { 0, 2, 1, 4, 0, -1 }, // ' '
00104      { 1, 2, 9, 4, 1, -9 }, // '!'
00105      { 4, 3, 3, 5, 1, -9 }, // '"'
00106      { 6, 8, 9, 8, 0, -9 }, // '#'
00107      { 15, 6, 11, 8, 1, -10 }, // '$'
00108      { 24, 10, 9, 12, 1, -9 }, // '%'
00109      { 36, 9, 9, 11, 1, -9 }, // '&'
00110      { 47, 2, 3, 4, 1, -9 }, // "'"
00111      { 48, 5, 11, 6, 1, -9 }, // '('
00112      { 55, 5, 11, 6, 0, -9 }, // ')'
00113      { 62, 4, 4, 6, 1, -10 }, // '*'
00114      { 64, 8, 6, 8, 0, -7 }, // '+'
00115      { 70, 3, 3, 4, 1, -2 }, // ','
00116      { 72, 5, 1, 5, 0, -4 }, // '-'
00117      { 73, 2, 1, 4, 1, -1 }, // '.'
00118      { 74, 6, 11, 6, 0, -9 }, // '/'
00119      { 83, 7, 9, 8, 1, -9 }, // '0'
00120      { 91, 4, 9, 8, 1, -9 }, // '1'
00121      { 96, 7, 9, 8, 1, -9 }, // '2'
00122      { 104, 7, 9, 8, 1, -9 }, // '3'
00123      { 112, 8, 9, 8, 0, -9 }, // '4'
00124      { 121, 7, 9, 8, 0, -9 }, // '5'
00125      { 129, 7, 9, 8, 1, -9 }, // '6'
00126      { 137, 6, 9, 8, 1, -9 }, // '7'
00127      { 144, 7, 9, 8, 1, -9 }, // '8'
00128      { 152, 7, 9, 8, 1, -9 }, // '9'
00129      { 160, 2, 6, 4, 1, -6 }, // ':'
00130      { 162, 3, 7, 4, 0, -6 }, // ';'
00131      { 165, 7, 7, 9, 1, -7 }, // '<'
00132      { 172, 7, 3, 8, 1, -5 }, // '='
00133      { 175, 7, 7, 9, 1, -7 }, // '>'
00134      { 182, 6, 9, 8, 1, -9 }, // '?'
00135      { 189, 10, 9, 12, 1, -9 }, // '@'
00136      { 201, 10, 9, 10, 0, -9 }, // 'A'
00137      { 213, 7, 9, 9, 1, -9 }, // 'B'
00138      { 221, 10, 9, 11, 1, -9 }, // 'C'
00139      { 233, 9, 9, 11, 1, -9 }, // 'D'
00140      { 244, 6, 9, 8, 1, -9 }, // 'E'
00141      { 251, 6, 9, 7, 1, -9 }, // 'F'
00142      { 258, 11, 9, 12, 1, -9 }, // 'G'
00143      { 271, 7, 9, 9, 1, -9 }, // 'H'
00144      { 279, 2, 9, 4, 1, -9 }, // 'I'
00145      { 282, 6, 9, 7, 0, -9 }, // 'J'
00146      { 289, 7, 9, 8, 1, -9 }, // 'K'
00147      { 297, 6, 9, 7, 1, -9 }, // 'L'
00148      { 304, 10, 9, 12, 1, -9 }, // 'M'
00149      { 316, 8, 9, 10, 1, -9 }, // 'N'
00150      { 325, 10, 9, 12, 1, -9 }, // 'O'
00151      { 337, 7, 9, 9, 1, -9 }, // 'P'
00152      { 345, 11, 9, 13, 1, -9 }, // 'Q'
00153      { 358, 7, 9, 9, 1, -9 }, // 'R'
00154      { 366, 6, 9, 8, 1, -9 }, // 'S'
00155      { 373, 6, 9, 6, 0, -9 }, // 'T'
00156      { 380, 7, 9, 9, 1, -9 }, // 'U'
00157      { 388, 9, 9, 9, 0, -9 }, // 'V'
00158      { 399, 12, 9, 13, 0, -9 }, // 'W'
00159      { 413, 8, 9, 8, 0, -9 }, // 'X'
00160      { 422, 8, 9, 8, 0, -9 }, // 'Y'
00161      { 431, 7, 9, 8, 1, -9 }, // 'Z'
00162      { 439, 4, 11, 5, 1, -9 }, // '['
00163      { 445, 6, 9, 8, 1, -9 }, // '\'
00164      { 452, 4, 11, 5, 0, -9 }, // ']'
00165      { 458, 8, 7, 8, 0, -9 }, // '^'
00166      { 465, 7, 1, 7, 0, 1 }, // '_'
00167      { 466, 4, 2, 6, 1, -10 }, // '`'
00168      { 467, 8, 7, 10, 1, -7 }, // 'a'
00169      { 474, 8, 9, 10, 1, -9 }, // 'b'
00170      { 483, 8, 7, 9, 1, -7 }, // 'c'
00171      { 490, 8, 9, 10, 1, -9 }, // 'd'
00172      { 499, 8, 7, 10, 1, -7 }, // 'e'
00173      { 506, 5, 9, 5, 0, -9 }, // 'f'
00174      { 512, 8, 9, 10, 1, -7 }, // 'g'
00175      { 521, 7, 9, 9, 1, -9 }, // 'h'

```



```

00176     { 529, 2, 9, 4, 1, -9 }, // 'i'
00177     { 532, 3, 11, 4, 0, -9 }, // 'j'
00178     { 537, 6, 9, 7, 1, -9 }, // 'k'
00179     { 544, 2, 9, 4, 1, -9 }, // 'l'
00180     { 547, 10, 7, 12, 1, -7 }, // 'm'
00181     { 556, 7, 7, 9, 1, -7 }, // 'n'
00182     { 563, 8, 7, 10, 1, -7 }, // 'o'
00183     { 570, 8, 9, 10, 1, -7 }, // 'p'
00184     { 579, 8, 9, 10, 1, -7 }, // 'q'
00185     { 588, 4, 7, 5, 1, -7 }, // 'r'
00186     { 592, 5, 7, 7, 1, -7 }, // 's'
00187     { 597, 4, 9, 6, 1, -9 }, // 't'
00188     { 602, 7, 7, 9, 1, -7 }, // 'u'
00189     { 609, 8, 7, 8, 0, -7 }, // 'v'
00190     { 616, 11, 7, 11, 0, -7 }, // 'w'
00191     { 626, 7, 7, 7, 0, -7 }, // 'x'
00192     { 633, 7, 9, 7, 0, -7 }, // 'y'
00193     { 641, 6, 7, 7, 1, -7 }, // 'z'
00194     { 647, 4, 11, 5, 0, -9 }, // '{'
00195     { 653, 2, 9, 10, 4, -9 }, // '|'
00196     { 656, 4, 11, 5, 0, -9 } // '}'
00197 };
00198 const GFXfont URW_Gothic_L_Book_12 PROGMEM = {
00199 (uint8_t *)URW_Gothic_L_Book_12Bitmaps, (GFXglyph *)URW_Gothic_L_Book_12Glyphs, 0x20, 0x7E, 17};

```

## 2.10 gothic9.h

```

00001 // Created by https://oledisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t URW_Gothic_L_Book_9Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0x88, // '!'
00008     0xD8, // '"'
00009     0x51, 0x4F, 0x94, 0xFA, 0x8A, 0x00, // '#'
00010     0x45, 0xA0, 0xC1, 0x49, 0x80, // '$'
00011     0xC8, 0xB0, 0xB0, 0x60, 0x2C, 0x32, 0x1C, // '%'
00012     0x70, 0xA1, 0xD3, 0x49, 0x91, 0x1D, 0x00, // '&'
00013     0xA0, // "'"
00014     0x04, 0x88, 0x88, 0x42, // '('
00015     0x04, 0x22, 0x22, 0x48, // ')'
00016     0xDB, 0x00, // '*'
00017     0x20, 0x8F, 0x88, 0x20, // '+'
00018     0x20, // ','
00019     0xE0, // '-'
00020     0x80, // '.'
00021     0x21, 0x08, 0x84, 0x22, 0x10, // '/'
00022     0x72, 0x28, 0xA2, 0x8A, 0x27, 0x00, // '0'
00023     0xC9, 0x24, 0x90, // '1'
00024     0x72, 0x20, 0x84, 0x21, 0x0F, 0x80, // '2'
00025     0x64, 0x84, 0x41, 0x49, 0x80, // '3'
00026     0x08, 0x30, 0xA2, 0x44, 0x9F, 0x82, 0x00, // '4'
00027     0x72, 0x07, 0x02, 0x0A, 0x27, 0x00, // '5'
00028     0x21, 0x07, 0x22, 0x8A, 0x27, 0x00, // '6'
00029     0xF0, 0x88, 0x44, 0x21, 0x00, // '7'
00030     0x64, 0xA4, 0xC9, 0x49, 0x80, // '8'
00031     0x72, 0x28, 0xA2, 0x70, 0x82, 0x00, // '9'
00032     0x80, 0x80, // ':'
00033     0x40, 0x28, // ';'
00034     0x19, 0x88, 0x1C, 0x08, // '<'
00035     0xF8, 0x0F, 0x80, // '='
00036     0x81, 0x81, 0x98, 0x80, // '>'
00037     0x72, 0x20, 0x84, 0x20, 0x02, 0x00, // '?'
00038     0x3C, 0x52, 0xAA, 0xCA, 0xD2, 0x6C, 0x38, // '@'
00039     0x10, 0x30, 0x28, 0x48, 0x7C, 0x84, 0x84, // 'A'
00040     0xE4, 0xA5, 0xC9, 0x4B, 0xC0, // 'B'
00041     0x3C, 0x42, 0x80, 0x80, 0x80, 0x42, 0x3C, // 'C'
00042     0xF1, 0x12, 0x14, 0x28, 0x51, 0x3C, 0x00, // 'D'
00043     0xF4, 0x21, 0xE8, 0x43, 0xC0, // 'E'
00044     0xF4, 0x21, 0xE8, 0x42, 0x00, // 'F'
00045     0x38, 0x44, 0x80, 0x80, 0x9E, 0xC2, 0x3C, // 'G'
00046     0x8A, 0x28, 0xBE, 0x8A, 0x28, 0x80, // 'H'
00047     0xAA, 0xA8, // 'I'
00048     0x10, 0x84, 0x21, 0x49, 0x80, // 'J'
00049     0x8A, 0x4A, 0x30, 0xA2, 0x48, 0x80, // 'K'
00050     0x84, 0x21, 0x08, 0x43, 0xC0, // 'L'
00051     0xC6, 0xC6, 0xCA, 0xAA, 0xAA, 0xB2, 0x92, // 'M'
00052     0x8B, 0x2C, 0xAA, 0xAA, 0x68, 0x80, // 'N'
00053     0x38, 0x44, 0x82, 0x82, 0x82, 0x44, 0x38, // 'O'
00054     0xF2, 0x28, 0xA2, 0xF2, 0x08, 0x00, // 'P'
00055     0x38, 0x44, 0x82, 0x82, 0xF2, 0x8C, 0x7E, // 'Q'
00056     0xF2, 0x28, 0xA6, 0xA2, 0x49, 0x80, // 'R'

```

```

00057      0x64,0xA0,0xC1,0x4B,0x80, // 'S'
00058      0xF8,0x82,0x08,0x20,0x82,0x00, // 'T'
00059      0x8A,0x28,0xA2,0x8A,0x27,0x00, // 'U'
00060      0x85,0x09,0x22,0x43,0x06,0x04,0x00, // 'V'
00061      0x89,0x22,0x49,0x91,0x54,0x56,0x19,0x82,0x20, // 'W'
00062      0x89,0x43,0x08,0x51,0x48,0x80, // 'X'
00063      0x8A,0x25,0x14,0x20,0x82,0x00, // 'Y'
00064      0xF0,0x88,0x84,0x43,0xC0, // 'Z'
00065      0x1A,0x49,0x22, // '['
00066      0x88,0x84,0x42,0x20, // '\
00067      0x19,0x24,0x94, // ']'
00068      0x21,0x45,0x12,0x88, // '^'
00069      0xF8, // '._'
00070      0x80, // '``'
00071      0x7A,0x28,0xA2,0x78, // 'a'
00072      0x82,0x0F,0x22,0x8A,0x2F,0x00, // 'b'
00073      0x7A,0x08,0x20,0x78, // 'c'
00074      0x08,0x27,0xA2,0x8A,0x27,0x80, // 'd'
00075      0x72,0x2F,0xA0,0x78, // 'e'
00076      0x24,0xE4,0x44,0x40, // 'f'
00077      0x7A,0x28,0xA2,0x7A,0x27,0x00, // 'g'
00078      0x82,0x0F,0x22,0x8A,0x28,0x80, // 'h'
00079      0x8A,0xA8, // 'i'
00080      0x41,0x24,0x92,0x80, // 'j'
00081      0x84,0x25,0x4C,0x52,0x40, // 'k'
00082      0xAA,0xA8, // 'l'
00083      0xEC,0x92,0x92,0x92,0x92, // 'm'
00084      0xF2,0x28,0xA2,0x88, // 'n'
00085      0x72,0x28,0xA2,0x70, // 'o'
00086      0xF2,0x28,0xA2,0xF2,0x08,0x00, // 'p'
00087      0x7A,0x28,0xA2,0x78,0x20,0x80, // 'q'
00088      0xD2,0x48, // 'r'
00089      0xE8,0x4A,0xE0, // 's'
00090      0x44,0xE4,0x44,0x40, // 't'
00091      0x8A,0x28,0xA2,0x78, // 'u'
00092      0x8A,0x45,0x18,0x20, // 'v'
00093      0x82,0x92,0x6C,0x6C,0x24, // 'w'
00094      0x93,0x08,0xC9,0x00, // 'x'
00095      0x8A,0x45,0x18,0x20,0x84,0x00, // 'y'
00096      0xF1,0x11,0x0F,0x00, // 'z'
00097      0x64,0x4C,0x44,0x42, // '{'
00098      0xAA,0xA8, // '|
00099      0xC4,0x46,0x44,0x48 // '}'
00100  };
00101  const GFXglyph URW_Gothic_L_Book_9Glyphs[] PROGMEM = {
00102  // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103      { 0, 2, 1, 4, 0, -1 }, // ' '
00104      { 1, 2, 7, 4, 1, -7 }, // '!'
00105      { 3, 3, 2, 5, 1, -7 }, // '"'
00106      { 4, 6, 7, 6, 0, -7 }, // '#'
00107      { 10, 5, 7, 7, 1, -7 }, // '$'
00108      { 15, 8, 7, 10, 1, -7 }, // '%'
00109      { 22, 7, 7, 8, 1, -7 }, // '&'
00110      { 29, 2, 2, 4, 1, -7 }, // "'"
00111      { 30, 4, 8, 5, 1, -7 }, // '('
00112      { 34, 4, 8, 5, 0, -7 }, // ')'
00113      { 38, 3, 3, 5, 1, -7 }, // '*'
00114      { 40, 6, 5, 6, 0, -5 }, // '+'
00115      { 44, 2, 2, 4, 1, -1 }, // ','
00116      { 45, 4, 1, 4, 0, -3 }, // '-'
00117      { 46, 2, 1, 4, 1, -1 }, // '.'
00118      { 47, 5, 8, 5, 0, -7 }, // '/'
00119      { 52, 6, 7, 6, 1, -7 }, // '0'
00120      { 58, 3, 7, 6, 2, -7 }, // '1'
00121      { 61, 6, 7, 6, 0, -7 }, // '2'
00122      { 67, 5, 7, 6, 0, -7 }, // '3'
00123      { 72, 7, 7, 6, 0, -7 }, // '4'
00124      { 79, 6, 7, 6, 0, -7 }, // '5'
00125      { 85, 6, 7, 6, 1, -7 }, // '6'
00126      { 91, 5, 7, 6, 0, -7 }, // '7'
00127      { 96, 5, 7, 6, 1, -7 }, // '8'
00128      { 101, 6, 7, 6, 1, -7 }, // '9'
00129      { 107, 2, 5, 4, 1, -5 }, // ':'
00130      { 109, 3, 5, 4, 0, -5 }, // ';'
00131      { 111, 6, 5, 7, 1, -5 }, // '<'
00132      { 115, 6, 3, 6, 0, -4 }, // '='
00133      { 118, 6, 5, 7, 1, -5 }, // '>'
00134      { 122, 6, 7, 7, 0, -7 }, // '?'
00135      { 128, 8, 7, 10, 1, -7 }, // '@'
00136      { 135, 8, 7, 8, 0, -7 }, // 'A'
00137      { 142, 5, 7, 7, 1, -7 }, // 'B'
00138      { 147, 8, 7, 9, 1, -7 }, // 'C'
00139      { 154, 7, 7, 9, 1, -7 }, // 'D'
00140      { 161, 5, 7, 6, 1, -7 }, // 'E'
00141      { 166, 5, 7, 6, 1, -7 }, // 'F'
00142      { 171, 8, 7, 10, 1, -7 }, // 'G'
00143      { 178, 6, 7, 8, 1, -7 }, // 'H'

```

```

00144      { 184, 2, 7, 4, 1, -7 }, // 'I'
00145      { 186, 5, 7, 6, 0, -7 }, // 'J'
00146      { 191, 6, 7, 7, 1, -7 }, // 'K'
00147      { 197, 5, 7, 6, 1, -7 }, // 'L'
00148      { 202, 8, 7, 10, 1, -7 }, // 'M'
00149      { 209, 6, 7, 8, 1, -7 }, // 'N'
00150      { 215, 8, 7, 10, 1, -7 }, // 'O'
00151      { 222, 6, 7, 8, 1, -7 }, // 'P'
00152      { 228, 8, 7, 10, 1, -7 }, // 'Q'
00153      { 235, 6, 7, 8, 1, -7 }, // 'R'
00154      { 241, 5, 7, 7, 1, -7 }, // 'S'
00155      { 246, 6, 7, 6, 0, -7 }, // 'T'
00156      { 252, 6, 7, 8, 1, -7 }, // 'U'
00157      { 258, 7, 7, 7, 0, -7 }, // 'V'
00158      { 265, 10, 7, 10, 0, -7 }, // 'W'
00159      { 274, 6, 7, 6, 0, -7 }, // 'X'
00160      { 280, 6, 7, 6, 0, -7 }, // 'Y'
00161      { 286, 5, 7, 7, 1, -7 }, // 'Z'
00162      { 291, 3, 8, 5, 1, -7 }, // '['
00163      { 294, 4, 7, 6, 1, -7 }, // '\'
00164      { 298, 3, 8, 5, 1, -7 }, // ']'
00165      { 301, 6, 5, 6, 0, -7 }, // '^'
00166      { 305, 6, 1, 6, 0, 0 }, // '_'
00167      { 306, 3, 2, 4, 1, -7 }, // '`'
00168      { 307, 6, 5, 8, 1, -5 }, // 'a'
00169      { 311, 6, 7, 8, 1, -7 }, // 'b'
00170      { 317, 6, 5, 8, 1, -5 }, // 'c'
00171      { 321, 6, 7, 8, 1, -7 }, // 'd'
00172      { 327, 6, 5, 8, 1, -5 }, // 'e'
00173      { 331, 4, 7, 4, 0, -7 }, // 'f'
00174      { 335, 6, 7, 8, 1, -5 }, // 'g'
00175      { 341, 6, 7, 8, 1, -7 }, // 'h'
00176      { 347, 2, 7, 4, 1, -7 }, // 'i'
00177      { 349, 3, 9, 4, 0, -7 }, // 'j'
00178      { 353, 5, 7, 6, 1, -7 }, // 'k'
00179      { 358, 2, 7, 4, 1, -7 }, // 'l'
00180      { 360, 8, 5, 10, 1, -5 }, // 'm'
00181      { 365, 6, 5, 8, 1, -5 }, // 'n'
00182      { 369, 6, 5, 8, 1, -5 }, // 'o'
00183      { 373, 6, 7, 8, 1, -5 }, // 'p'
00184      { 379, 6, 7, 8, 1, -5 }, // 'q'
00185      { 385, 3, 5, 4, 1, -5 }, // 'r'
00186      { 387, 4, 5, 6, 1, -5 }, // 's'
00187      { 390, 4, 7, 4, 0, -7 }, // 't'
00188      { 394, 6, 5, 8, 1, -5 }, // 'u'
00189      { 398, 6, 5, 6, 0, -5 }, // 'v'
00190      { 402, 8, 5, 8, 0, -5 }, // 'w'
00191      { 407, 5, 5, 5, 0, -5 }, // 'x'
00192      { 411, 6, 7, 6, 0, -5 }, // 'y'
00193      { 417, 5, 5, 6, 1, -5 }, // 'z'
00194      { 421, 4, 8, 4, 0, -7 }, // '{'
00195      { 425, 2, 7, 8, 3, -7 }, // '|'
00196      { 427, 4, 8, 4, 0, -7 }, // '}'
00197 };
00198 const GFXfont URW_Gothic_L_Book_9 PROGMEM = {
00199 (uint8_t *)URW_Gothic_L_Book_9Bitmaps, (GFXglyph *)URW_Gothic_L_Book_9Glyphs, 0x20, 0x7E, 13};

```

## 2.11 mono12.h

```

00001 // Created by https://oledisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Monospaced_plain_12Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0xA2, 0x80, // '!'
00008     0xAA, 0xA0, // '"'
00009     0x14, 0x24, 0x7E, 0x28, 0x28, 0xFC, 0x48, 0x50, // '#'
00010     0x21, 0xCA, 0xA8, 0xE0, 0xE2, 0xAA, 0x70, 0x82, 0x00, // '$'
00011     0x60, 0x90, 0x90, 0x64, 0x18, 0x6C, 0x12, 0x12, 0x0C, // '%'
00012     0x38, 0x81, 0x03, 0x06, 0x12, 0xA7, 0x64, 0x74, // '&'
00013     0xA8, // '''
00014     0x64, 0x48, 0x88, 0x88, 0x44, 0x60, // '('
00015     0xC4, 0x42, 0x22, 0x22, 0x44, 0xC0, // ')'
00016     0x22, 0xA7, 0x1C, 0xA8, 0x80, // '*'
00017     0x10, 0x10, 0x10, 0xFE, 0x10, 0x10, 0x10, // '+'
00018     0x4A, 0x00, // ','
00019     0xE0, // '-'
00020     0xA0, // '.'
00021     0x04, 0x10, 0x20, 0x81, 0x04, 0x08, 0x20, 0x41, 0x00, // '/'
00022     0x78, 0x92, 0x14, 0x29, 0x50, 0xA1, 0x24, 0x78, // '0'
00023     0xE0, 0x82, 0x08, 0x20, 0x82, 0x08, 0xF8, // '1'
00024     0x79, 0x08, 0x10, 0x20, 0x82, 0x08, 0x20, 0xFC, // '2'

```

```

00025     0x79, 0x08, 0x10, 0x23, 0x80, 0x81, 0x42, 0x78, // '3'
00026     0x18, 0x30, 0xA3, 0x44, 0x91, 0x3F, 0x04, 0x08, // '4'
00027     0xF9, 0x02, 0x07, 0xC0, 0xC0, 0x81, 0x46, 0x78, // '5'
00028     0x38, 0x8A, 0x05, 0xCC, 0xD0, 0xA1, 0x26, 0x78, // '6'
00029     0xFC, 0x18, 0x20, 0x41, 0x02, 0x08, 0x10, 0x40, // '7'
00030     0x79, 0x0A, 0x14, 0x27, 0x90, 0xA1, 0x42, 0x78, // '8'
00031     0x79, 0x92, 0x14, 0x28, 0xCE, 0x81, 0x44, 0x70, // '9'
00032     0xA0, 0xA0, // ':'
00033     0x48, 0x04, 0xA0, // ';'
00034     0x04, 0x73, 0x06, 0x03, 0x80, 0x80, // '<'
00035     0xFC, 0x03, 0xF0, // '='
00036     0x80, 0xE0, 0x30, 0x67, 0x10, 0x00, // '>'
00037     0x72, 0x20, 0x8C, 0x61, 0x00, 0x10, 0x40, // '?'
00038     0x38, 0x9A, 0x14, 0xEA, 0x54, 0xA7, 0x60, 0x40, 0x70, // '@'
00039     0x30, 0x60, 0xC2, 0x44, 0x89, 0x1E, 0x42, 0x84, // 'A'
00040     0xF9, 0x0A, 0x14, 0x2F, 0x90, 0xA1, 0x42, 0xF8, // 'B'
00041     0x38, 0x8A, 0x04, 0x08, 0x10, 0x20, 0x22, 0x38, // 'C'
00042     0xF1, 0x12, 0x14, 0x28, 0x50, 0xA1, 0x44, 0xF0, // 'D'
00043     0xFD, 0x02, 0x04, 0x0F, 0xD0, 0x20, 0x40, 0xFC, // 'E'
00044     0xFD, 0x02, 0x04, 0x0F, 0xD0, 0x20, 0x40, 0x80, // 'F'
00045     0x38, 0x8A, 0x04, 0x08, 0xD0, 0xA1, 0x22, 0x38, // 'G'
00046     0x85, 0x0A, 0x14, 0x2F, 0xD0, 0xA1, 0x42, 0x84, // 'H'
00047     0xF8, 0x82, 0x08, 0x20, 0x82, 0x08, 0xF8, // 'I'
00048     0x38, 0x20, 0x82, 0x08, 0x20, 0xA2, 0x70, // 'J'
00049     0x85, 0x12, 0x45, 0x0E, 0x12, 0x26, 0x44, 0x84, // 'K'
00050     0x81, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0xFC, // 'L'
00051     0x85, 0x9B, 0x35, 0xAB, 0x56, 0xA1, 0x42, 0x84, // 'M'
00052     0xC5, 0x8A, 0x95, 0x2B, 0x52, 0xA5, 0x46, 0x8C, // 'N'
00053     0x78, 0x92, 0x14, 0x28, 0x50, 0xA1, 0x24, 0x78, // 'O'
00054     0xF9, 0x0A, 0x14, 0x2F, 0x90, 0x20, 0x40, 0x80, // 'P'
00055     0x78, 0x92, 0x14, 0x28, 0x50, 0xA1, 0x26, 0x78, 0x10, 0x20, // 'Q'
00056     0xF8, 0x84, 0x84, 0x84, 0xF8, 0x88, 0x84, 0x84, 0x82, // 'R'
00057     0x79, 0x0A, 0x06, 0x07, 0x80, 0x81, 0x42, 0x78, // 'S'
00058     0xFE, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, // 'T'
00059     0x85, 0x0A, 0x14, 0x28, 0x50, 0xA1, 0x42, 0x78, // 'U'
00060     0x85, 0x09, 0x22, 0x44, 0x89, 0x0C, 0x18, 0x30, // 'V'
00061     0x82, 0x92, 0x92, 0xAA, 0xAA, 0xAA, 0x6C, 0x44, 0x44, // 'W'
00062     0x84, 0x91, 0x21, 0x83, 0x06, 0x12, 0x24, 0x84, // 'X'
00063     0x82, 0x44, 0x28, 0x28, 0x10, 0x10, 0x10, 0x10, 0x10, // 'Y'
00064     0xFC, 0x18, 0x20, 0x83, 0x04, 0x10, 0x60, 0xFC, // 'Z'
00065     0xD2, 0x49, 0x24, 0x93, 0x00, // '['
00066     0x80, 0x81, 0x01, 0x02, 0x02, 0x04, 0x04, 0x08, 0x08, // '\'
00067     0xC9, 0x24, 0x92, 0x4B, 0x00, // ']'
00068     0x30, 0x92, 0x10, // '^'
00069     0xFE, // '_'
00070     0x42, // '`'
00071     0x72, 0x20, 0x9E, 0x8A, 0x27, 0x80, // 'a'
00072     0x82, 0x08, 0x3C, 0x8A, 0x28, 0xA2, 0x8B, 0xC0, // 'b'
00073     0x73, 0x28, 0x20, 0x83, 0x07, 0x80, // 'c'
00074     0x08, 0x20, 0x9E, 0x8A, 0x28, 0xA2, 0x89, 0xE0, // 'd'
00075     0x73, 0x28, 0xBE, 0x82, 0x27, 0x00, // 'e'
00076     0x18, 0x82, 0x3E, 0x20, 0x82, 0x08, 0x20, 0x80, // 'f'
00077     0x7A, 0x28, 0xA2, 0x8A, 0x27, 0x82, 0x48, 0xC0, // 'g'
00078     0x82, 0x08, 0x2C, 0xCA, 0x28, 0xA2, 0x8A, 0x20, // 'h'
00079     0x20, 0x00, 0x38, 0x20, 0x82, 0x08, 0x23, 0xE0, // 'i'
00080     0x20, 0x0E, 0x22, 0x22, 0x22, 0x22, 0xC0, // 'j'
00081     0x82, 0x08, 0x22, 0x92, 0x8C, 0x28, 0x92, 0x20, // 'k'
00082     0xE0, 0x82, 0x08, 0x20, 0x82, 0x08, 0x20, 0x60, // 'l'
00083     0xFA, 0xAA, 0xAA, 0xAA, 0x80, // 'm'
00084     0xB3, 0x28, 0xA2, 0x8A, 0x28, 0x80, // 'n'
00085     0x72, 0x28, 0xA2, 0x8A, 0x27, 0x00, // 'o'
00086     0xF2, 0x28, 0xA2, 0x8A, 0x2F, 0x20, 0x82, 0x00, // 'p'
00087     0x7A, 0x28, 0xA2, 0x8A, 0x27, 0x82, 0x08, 0x20, // 'q'
00088     0xF3, 0x28, 0x20, 0x82, 0x08, 0x00, // 'r'
00089     0x72, 0x28, 0x1C, 0x0A, 0x27, 0x00, // 's'
00090     0x20, 0x8F, 0x88, 0x20, 0x82, 0x08, 0x38, // 't'
00091     0x8A, 0x28, 0xA2, 0x8A, 0x27, 0x80, // 'u'
00092     0x8A, 0x25, 0x14, 0x50, 0x82, 0x00, // 'v'
00093     0x82, 0x82, 0x54, 0x54, 0x6C, 0x28, 0x28, // 'w'
00094     0x89, 0x45, 0x08, 0x51, 0x48, 0x80, // 'x'
00095     0x8A, 0x25, 0x14, 0x51, 0x82, 0x08, 0x43, 0x00, // 'y'
00096     0xF8, 0x21, 0x08, 0x42, 0x0F, 0x80, // 'z'
00097     0x38, 0x82, 0x08, 0x23, 0x02, 0x08, 0x20, 0x83, 0x80, // '{'
00098     0xAA, 0xAA, 0xAA, // '|'
00099     0xE0, 0x82, 0x08, 0x20, 0x62, 0x08, 0x20, 0x8E, 0x00 // '}'
00100 };
00101 const GFXglyph Monospaced_plain_12Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103 { 0, 2, 1, 8, 0, -1 }, // ' '
00104 { 1, 2, 9, 8, 3, -9 }, // '!'
00105 { 4, 4, 3, 8, 2, -9 }, // '"'
00106 { 6, 8, 8, 8, 0, -8 }, // '#'
00107 { 14, 6, 11, 8, 1, -9 }, // '$'
00108 { 23, 8, 9, 8, 0, -9 }, // '%'
00109 { 32, 7, 9, 8, 1, -9 }, // '&'
00110 { 40, 2, 3, 8, 3, -9 }, // "'"
00111 { 41, 4, 11, 8, 3, -10 }, // '('

```

```

00112     { 47, 4, 11, 8, 2, -10 }, // ')'
00113     { 53, 6, 6, 8, 1, -9 }, // '*'
00114     { 58, 8, 7, 8, 0, -7 }, // '+'
00115     { 65, 3, 3, 8, 2, -2 }, // ','
00116     { 67, 4, 1, 8, 2, -4 }, // '-'
00117     { 68, 2, 2, 8, 3, -2 }, // '.'
00118     { 69, 7, 10, 8, 1, -9 }, // '/'
00119     { 78, 7, 9, 8, 1, -9 }, // '0'
00120     { 86, 6, 9, 8, 1, -9 }, // '1'
00121     { 93, 7, 9, 8, 1, -9 }, // '2'
00122     { 101, 7, 9, 8, 1, -9 }, // '3'
00123     { 109, 7, 9, 8, 1, -9 }, // '4'
00124     { 117, 7, 9, 8, 1, -9 }, // '5'
00125     { 125, 7, 9, 8, 1, -9 }, // '6'
00126     { 133, 7, 9, 8, 1, -9 }, // '7'
00127     { 141, 7, 9, 8, 1, -9 }, // '8'
00128     { 149, 7, 9, 8, 1, -9 }, // '9'
00129     { 157, 2, 6, 8, 3, -6 }, // ':'
00130     { 159, 3, 7, 8, 2, -6 }, // ';'
00131     { 162, 7, 6, 8, 1, -7 }, // '<'
00132     { 168, 7, 3, 8, 1, -5 }, // '='
00133     { 171, 7, 6, 8, 1, -7 }, // '>'
00134     { 177, 6, 9, 8, 2, -9 }, // '?'
00135     { 184, 7, 10, 8, 1, -8 }, // '@'
00136     { 193, 7, 9, 8, 1, -9 }, // 'A'
00137     { 201, 7, 9, 8, 1, -9 }, // 'B'
00138     { 209, 7, 9, 8, 1, -9 }, // 'C'
00139     { 217, 7, 9, 8, 1, -9 }, // 'D'
00140     { 225, 7, 9, 8, 1, -9 }, // 'E'
00141     { 233, 7, 9, 8, 1, -9 }, // 'F'
00142     { 241, 7, 9, 8, 1, -9 }, // 'G'
00143     { 249, 7, 9, 8, 1, -9 }, // 'H'
00144     { 257, 6, 9, 8, 1, -9 }, // 'I'
00145     { 264, 6, 9, 8, 1, -9 }, // 'J'
00146     { 271, 7, 9, 8, 1, -9 }, // 'K'
00147     { 279, 7, 9, 8, 1, -9 }, // 'L'
00148     { 287, 7, 9, 8, 1, -9 }, // 'M'
00149     { 295, 7, 9, 8, 1, -9 }, // 'N'
00150     { 303, 7, 9, 8, 1, -9 }, // 'O'
00151     { 311, 7, 9, 8, 1, -9 }, // 'P'
00152     { 319, 7, 11, 8, 1, -9 }, // 'Q'
00153     { 329, 8, 9, 8, 1, -9 }, // 'R'
00154     { 338, 7, 9, 8, 1, -9 }, // 'S'
00155     { 346, 8, 9, 8, 0, -9 }, // 'T'
00156     { 355, 7, 9, 8, 1, -9 }, // 'U'
00157     { 363, 7, 9, 8, 1, -9 }, // 'V'
00158     { 371, 8, 9, 8, 0, -9 }, // 'W'
00159     { 380, 7, 9, 8, 1, -9 }, // 'X'
00160     { 388, 8, 9, 8, 0, -9 }, // 'Y'
00161     { 397, 7, 9, 8, 1, -9 }, // 'Z'
00162     { 405, 3, 11, 8, 3, -10 }, // '['
00163     { 410, 7, 10, 8, 1, -9 }, // '\'
00164     { 419, 3, 11, 8, 2, -10 }, // ']'
00165     { 424, 7, 3, 8, 0, -9 }, // '^'
00166     { 427, 8, 1, 8, 0, 2 }, // '_'
00167     { 428, 4, 2, 8, 2, -10 }, // '`'
00168     { 429, 6, 7, 8, 1, -7 }, // 'a'
00169     { 435, 6, 10, 8, 1, -10 }, // 'b'
00170     { 443, 6, 7, 8, 1, -7 }, // 'c'
00171     { 449, 6, 10, 8, 1, -10 }, // 'd'
00172     { 457, 6, 7, 8, 1, -7 }, // 'e'
00173     { 463, 6, 10, 8, 1, -10 }, // 'f'
00174     { 471, 6, 10, 8, 1, -7 }, // 'g'
00175     { 479, 6, 10, 8, 1, -10 }, // 'h'
00176     { 487, 6, 10, 8, 1, -10 }, // 'i'
00177     { 495, 4, 13, 8, 2, -10 }, // 'j'
00178     { 502, 6, 10, 8, 1, -10 }, // 'k'
00179     { 510, 6, 10, 8, 1, -10 }, // 'l'
00180     { 518, 6, 7, 8, 1, -7 }, // 'm'
00181     { 524, 6, 7, 8, 1, -7 }, // 'n'
00182     { 530, 6, 7, 8, 1, -7 }, // 'o'
00183     { 536, 6, 10, 8, 1, -7 }, // 'p'
00184     { 544, 6, 10, 8, 1, -7 }, // 'q'
00185     { 552, 6, 7, 8, 2, -7 }, // 'r'
00186     { 558, 6, 7, 8, 1, -7 }, // 's'
00187     { 564, 6, 9, 8, 1, -9 }, // 't'
00188     { 571, 6, 7, 8, 1, -7 }, // 'u'
00189     { 577, 6, 7, 8, 1, -7 }, // 'v'
00190     { 583, 8, 7, 8, 0, -7 }, // 'w'
00191     { 590, 6, 7, 8, 1, -7 }, // 'x'
00192     { 596, 6, 10, 8, 1, -7 }, // 'y'
00193     { 604, 6, 7, 8, 1, -7 }, // 'z'
00194     { 610, 6, 11, 8, 1, -10 }, // '{'
00195     { 619, 2, 12, 8, 3, -10 }, // '|'
00196     { 622, 6, 11, 8, 1, -10 }, // '}'
00197 };
00198 const GFXfont Monospaced_plain_12 PROGMEM = {

```

```
00199 (uint8_t *)Monospaced_plain_12Bitmaps, (GFXglyph *)Monospaced_plain_12Glyphs, 0x20, 0x7E, 15};
```

## 2.12 mono9.h

```
00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Monospaced_plain_9Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0x88, // '!'
00008     0xAA, // '"'
00009     0x31, 0x4F, 0x94, 0xF9, 0x8A, 0x00, // '#'
00010     0x21, 0xEA, 0x38, 0x38, 0xAF, 0x08, // '$'
00011     0xE2, 0x8E, 0x8C, 0x78, 0xA3, 0x80, // '%'
00012     0x72, 0x10, 0xCB, 0x51, 0xC0, // '&'
00013     0xA0, // "'"
00014     0x52, 0x49, 0x22, // '('
00015     0x91, 0x24, 0xA4, // ')'
00016     0xA9, 0xC7, 0x2A, // '*'
00017     0x20, 0x8F, 0x88, 0x20, // '+'
00018     0xA8, // ','
00019     0xC0, // '-'
00020     0x80, // '.'
00021     0x11, 0x08, 0xC4, 0x22, 0x00, // '/'
00022     0x64, 0xA5, 0x69, 0x49, 0x80, // '0'
00023     0xC4, 0x44, 0x44, 0xE0, // '1'
00024     0x64, 0x84, 0x66, 0x63, 0xC0, // '2'
00025     0x64, 0x84, 0xC1, 0x0B, 0x80, // '3'
00026     0x21, 0x18, 0xCA, 0x78, 0x80, // '4'
00027     0xF4, 0x21, 0xC1, 0x0B, 0x80, // '5'
00028     0x76, 0x21, 0xE9, 0x49, 0x80, // '6'
00029     0xF0, 0x88, 0x42, 0x31, 0x00, // '7'
00030     0x64, 0xA4, 0xC9, 0x49, 0x80, // '8'
00031     0x64, 0xA5, 0xE1, 0x1B, 0x80, // '9'
00032     0x80, 0x80, // ':'
00033     0x80, 0xA8, // ';'
00034     0x09, 0xCE, 0x06, // '<'
00035     0xF8, 0x0F, 0x80, // '='
00036     0x81, 0xC3, 0xB0, // '>'
00037     0xF0, 0x88, 0x84, 0x01, 0x00, // '?'
00038     0x64, 0xAD, 0x6B, 0x61, 0x80, // '@'
00039     0x63, 0x19, 0x29, 0x7A, 0x40, // 'A'
00040     0xE4, 0xA5, 0xC9, 0x4B, 0xC0, // 'B'
00041     0x74, 0x21, 0x08, 0x41, 0xC0, // 'C'
00042     0xE4, 0xA5, 0x29, 0x4B, 0x80, // 'D'
00043     0xF4, 0x21, 0xE8, 0x43, 0xC0, // 'E'
00044     0xF4, 0x21, 0xE8, 0x42, 0x00, // 'F'
00045     0x74, 0x21, 0x69, 0x49, 0xC0, // 'G'
00046     0x94, 0xA5, 0xE9, 0x4A, 0x40, // 'H'
00047     0xE4, 0x44, 0x44, 0xE0, // 'I'
00048     0x70, 0x84, 0x21, 0x0B, 0xC0, // 'J'
00049     0x95, 0x31, 0x8A, 0x52, 0x40, // 'K'
00050     0x84, 0x21, 0x08, 0x43, 0xC0, // 'L'
00051     0x97, 0xBD, 0xEF, 0x4A, 0x40, // 'M'
00052     0x96, 0xB5, 0x6B, 0x5A, 0x40, // 'N'
00053     0x64, 0xA5, 0x29, 0x49, 0x80, // 'O'
00054     0xE4, 0xA5, 0xC8, 0x42, 0x00, // 'P'
00055     0x64, 0xA5, 0x29, 0x49, 0x82, // 'Q'
00056     0xE2, 0x49, 0x38, 0xB2, 0x48, 0x80, // 'R'
00057     0x64, 0xA0, 0xE1, 0x49, 0x80, // 'S'
00058     0xF8, 0x82, 0x08, 0x20, 0x82, 0x00, // 'T'
00059     0x94, 0xA5, 0x29, 0x49, 0x80, // 'U'
00060     0x94, 0x98, 0xC6, 0x31, 0x80, // 'V'
00061     0x8A, 0x2A, 0xB6, 0x51, 0x45, 0x00, // 'W'
00062     0x94, 0x98, 0xC6, 0x4A, 0x40, // 'X'
00063     0x89, 0x45, 0x08, 0x20, 0x82, 0x00, // 'Y'
00064     0xF0, 0x88, 0x44, 0x43, 0xC0, // 'Z'
00065     0xD2, 0x49, 0x26, // '['
00066     0x82, 0x10, 0xC2, 0x10, 0x40, // '\'
00067     0xC9, 0x24, 0x96, // ']'
00068     0x23, 0x40, // '^'
00069     0xF8, // '_'
00070     0x88, // '`'
00071     0xE0, 0xBD, 0x2F, 0x00, // 'a'
00072     0x84, 0x21, 0xC9, 0x4A, 0x5C, // 'b'
00073     0x74, 0x21, 0x07, 0x00, // 'c'
00074     0x10, 0x84, 0xE9, 0x4A, 0x4E, // 'd'
00075     0x64, 0xBD, 0x07, 0x00, // 'e'
00076     0x32, 0x11, 0xE4, 0x21, 0x08, // 'f'
00077     0x74, 0xA5, 0x27, 0x09, 0x80, // 'g'
00078     0x84, 0x21, 0xE9, 0x4A, 0x52, // 'h'
00079     0x20, 0x00, 0x18, 0x20, 0x82, 0x3E, // 'i'
```

```

00080      0x20,0x06,0x22,0x22,0x2E, // 'j'
00081      0x84,0x21,0x2A,0x62,0x92, // 'k'
00082      0xE0,0x82,0x08,0x20,0x82,0x06, // 'l'
00083      0xFA,0xAA,0xAA,0xA8, // 'm'
00084      0xF4,0xA5,0x29,0x00, // 'n'
00085      0x64,0xA5,0x26,0x00, // 'o'
00086      0xE4,0xA5,0x2E,0x42,0x00, // 'p'
00087      0x74,0xA5,0x27,0x08,0x40, // 'q'
00088      0xE8,0x88,0x80, // 'r'
00089      0xF4,0x1C,0x2F,0x00, // 's'
00090      0x47,0x90,0x84,0x38, // 't'
00091      0x94,0xA5,0x2F,0x00, // 'u'
00092      0x94,0x98,0xC6,0x00, // 'v'
00093      0x8A,0xA5,0x14,0x50, // 'w'
00094      0x93,0x18,0xC9,0x00, // 'x'
00095      0x94,0x98,0xC4,0x23,0x00, // 'y'
00096      0xF1,0x18,0x8F,0x00, // 'z'
00097      0x64,0x48,0x44,0x46, // '{'
00098      0xAA,0xAA,0x80, // '|'
00099      0xC4,0x42,0x44,0x4C // '}'
00100  };
00101  const GFXglyph Monospaced_plain_9Glyphs[] PROGMEM = {
00102  // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103      { 0, 2, 1, 6, 0, -1 }, // ' '
00104      { 1, 2, 7, 6, 2, -7 }, // '!'
00105      { 3, 4, 2, 6, 1, -7 }, // '"'
00106      { 4, 6, 7, 6, 0, -7 }, // '#'
00107      { 10, 6, 8, 6, 1, -7 }, // '$'
00108      { 16, 6, 7, 6, 0, -7 }, // '%'
00109      { 22, 5, 7, 6, 1, -7 }, // '&'
00110      { 27, 2, 2, 6, 2, -7 }, // "'"
00111      { 28, 3, 8, 6, 2, -8 }, // '('
00112      { 31, 3, 8, 6, 2, -8 }, // ')'
00113      { 34, 6, 4, 6, 0, -7 }, // '*'
00114      { 37, 6, 5, 6, 0, -5 }, // '+'
00115      { 41, 2, 3, 6, 2, -1 }, // ','
00116      { 42, 3, 1, 6, 2, -3 }, // '-'
00117      { 43, 2, 1, 6, 2, -1 }, // '.'
00118      { 44, 5, 7, 6, 0, -7 }, // '/'
00119      { 49, 5, 7, 6, 1, -7 }, // '0'
00120      { 54, 4, 7, 6, 2, -7 }, // '1'
00121      { 58, 5, 7, 6, 1, -7 }, // '2'
00122      { 63, 5, 7, 6, 1, -7 }, // '3'
00123      { 68, 5, 7, 6, 1, -7 }, // '4'
00124      { 73, 5, 7, 6, 1, -7 }, // '5'
00125      { 78, 5, 7, 6, 1, -7 }, // '6'
00126      { 83, 5, 7, 6, 1, -7 }, // '7'
00127      { 88, 5, 7, 6, 1, -7 }, // '8'
00128      { 93, 5, 7, 6, 1, -7 }, // '9'
00129      { 98, 2, 5, 6, 2, -5 }, // ':'
00130      { 100, 2, 7, 6, 2, -5 }, // ';'
00131      { 102, 6, 4, 6, 1, -5 }, // '<'
00132      { 105, 6, 3, 6, 0, -4 }, // '='
00133      { 108, 6, 4, 6, 1, -5 }, // '>'
00134      { 111, 5, 7, 6, 1, -7 }, // '?'
00135      { 116, 5, 7, 6, 1, -6 }, // '@'
00136      { 121, 5, 7, 6, 1, -7 }, // 'A'
00137      { 126, 5, 7, 6, 1, -7 }, // 'B'
00138      { 131, 5, 7, 6, 1, -7 }, // 'C'
00139      { 136, 5, 7, 6, 1, -7 }, // 'D'
00140      { 141, 5, 7, 6, 1, -7 }, // 'E'
00141      { 146, 5, 7, 6, 1, -7 }, // 'F'
00142      { 151, 5, 7, 6, 1, -7 }, // 'G'
00143      { 156, 5, 7, 6, 1, -7 }, // 'H'
00144      { 161, 4, 7, 6, 1, -7 }, // 'I'
00145      { 165, 5, 7, 6, 1, -7 }, // 'J'
00146      { 170, 5, 7, 6, 1, -7 }, // 'K'
00147      { 175, 5, 7, 6, 1, -7 }, // 'L'
00148      { 180, 5, 7, 6, 1, -7 }, // 'M'
00149      { 185, 5, 7, 6, 1, -7 }, // 'N'
00150      { 190, 5, 7, 6, 1, -7 }, // 'O'
00151      { 195, 5, 7, 6, 1, -7 }, // 'P'
00152      { 200, 5, 8, 6, 1, -7 }, // 'Q'
00153      { 205, 6, 7, 6, 1, -7 }, // 'R'
00154      { 211, 5, 7, 6, 1, -7 }, // 'S'
00155      { 216, 6, 7, 6, 0, -7 }, // 'T'
00156      { 222, 5, 7, 6, 1, -7 }, // 'U'
00157      { 227, 5, 7, 6, 1, -7 }, // 'V'
00158      { 232, 6, 7, 6, 0, -7 }, // 'W'
00159      { 238, 5, 7, 6, 1, -7 }, // 'X'
00160      { 243, 6, 7, 6, 0, -7 }, // 'Y'
00161      { 249, 5, 7, 6, 1, -7 }, // 'Z'
00162      { 254, 3, 8, 6, 2, -8 }, // '['
00163      { 257, 5, 7, 6, 0, -7 }, // '\'
00164      { 262, 3, 8, 6, 2, -8 }, // ']'
00165      { 265, 6, 2, 6, 0, -7 }, // '^'
00166      { 267, 6, 1, 6, 0, 1 }, // '_'

```

```

00167     { 268, 3, 2, 6, 1, -8 }, // 'a'
00168     { 269, 5, 5, 6, 1, -5 }, // 'a'
00169     { 273, 5, 8, 6, 1, -8 }, // 'b'
00170     { 278, 5, 5, 6, 1, -5 }, // 'c'
00171     { 282, 5, 8, 6, 1, -8 }, // 'd'
00172     { 287, 5, 5, 6, 1, -5 }, // 'e'
00173     { 291, 5, 8, 6, 1, -8 }, // 'f'
00174     { 296, 5, 7, 6, 1, -5 }, // 'g'
00175     { 301, 5, 8, 6, 1, -8 }, // 'h'
00176     { 306, 6, 8, 6, 0, -8 }, // 'i'
00177     { 312, 4, 10, 6, 1, -8 }, // 'j'
00178     { 317, 5, 8, 6, 1, -8 }, // 'k'
00179     { 322, 6, 8, 6, 0, -8 }, // 'l'
00180     { 328, 6, 5, 6, 1, -5 }, // 'm'
00181     { 332, 5, 5, 6, 1, -5 }, // 'n'
00182     { 336, 5, 5, 6, 1, -5 }, // 'o'
00183     { 340, 5, 7, 6, 1, -5 }, // 'p'
00184     { 345, 5, 7, 6, 1, -5 }, // 'q'
00185     { 350, 4, 5, 6, 2, -5 }, // 'r'
00186     { 353, 5, 5, 6, 1, -5 }, // 's'
00187     { 357, 5, 6, 6, 1, -6 }, // 't'
00188     { 361, 5, 5, 6, 1, -5 }, // 'u'
00189     { 365, 5, 5, 6, 1, -5 }, // 'v'
00190     { 369, 6, 5, 6, 0, -5 }, // 'w'
00191     { 373, 5, 5, 6, 1, -5 }, // 'x'
00192     { 377, 5, 7, 6, 1, -5 }, // 'y'
00193     { 382, 5, 5, 6, 1, -5 }, // 'z'
00194     { 386, 4, 8, 6, 1, -8 }, // '{'
00195     { 390, 2, 9, 6, 2, -8 }, // '|'
00196     { 393, 4, 8, 6, 1, -8 } // '}'
00197 };
00198 const GFXfont Monospaced_plain_9 PROGMEM = {
00199 (uint8_t *)Monospaced_plain_9Bitmaps, (GFXglyph *)Monospaced_plain_9Glyphs, 0x20, 0x7E, 12};

```

## 2.13 opensans12.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Open_Sans-Regular_l2Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0xA2, 0x80, // '!'
00008     0xAA, 0xA0, // ' "'
00009     0x12, 0x0A, 0x05, 0x0F, 0xE2, 0x41, 0x23, 0xF8, 0x50, 0x48, 0x00, // '#'
00010     0x21, 0xEA, 0x28, 0x60, 0xE2, 0x8A, 0xF0, 0x80, // '$'
00011     0xC2, 0x52, 0x29, 0x15, 0xEA, 0xD7, 0xA8, 0x94, 0x4A, 0x47, 0x00, // '%'
00012     0x70, 0x24, 0x12, 0x0A, 0x06, 0x04, 0x92, 0x29, 0x08, 0x7A, 0x00, // '&'
00013     0xA8, // ' "'
00014     0x4C, 0x88, 0x88, 0x88, 0x8C, 0x40, // '('
00015     0x46, 0x22, 0x22, 0x22, 0x26, 0x40, // ')'
00016     0x20, 0x8F, 0x94, 0xD8, // '*'
00017     0x20, 0x82, 0x3E, 0x20, 0x82, 0x00, // '+'
00018     0x4A, 0x00, // ', '
00019     0xC0, // '-'
00020     0xA0, // '.'
00021     0x11, 0x08, 0x44, 0x21, 0x08, 0x80, // '/'
00022     0x72, 0x28, 0xA2, 0x8A, 0x28, 0xA2, 0x70, // '0'
00023     0x26, 0xA2, 0x22, 0x22, 0x20, // '1'
00024     0x72, 0x20, 0x82, 0x10, 0x84, 0x20, 0xF8, // '2'
00025     0xF0, 0x20, 0x86, 0x70, 0x20, 0x82, 0xF0, // '3'
00026     0x08, 0x18, 0x18, 0x28, 0x48, 0x48, 0xFE, 0x08, 0x08, // '4'
00027     0xFA, 0x08, 0x3C, 0x08, 0x20, 0x82, 0xF0, // '5'
00028     0x39, 0x08, 0x3C, 0x8A, 0x28, 0xA2, 0x70, // '6'
00029     0xF8, 0x20, 0x84, 0x10, 0x82, 0x10, 0x40, // '7'
00030     0x72, 0x28, 0xB6, 0x72, 0x28, 0xA2, 0xF0, // '8'
00031     0x72, 0x28, 0xA2, 0x89, 0xE0, 0x86, 0xE0, // '9'
00032     0xA0, 0xA0, // ':'
00033     0x48, 0x00, 0x94, // ';'
00034     0x08, 0xCC, 0x30, 0x30, 0x20, // '<'
00035     0xF8, 0x0F, 0x80, // '='
00036     0x81, 0x81, 0x86, 0x62, 0x00, // '>'
00037     0xF2, 0x20, 0x84, 0x20, 0x80, 0x18, 0x60, // '?'
00038     0x3E, 0x10, 0x49, 0xEA, 0x4A, 0xA2, 0xA8, 0xAA, 0x4A, 0x6C, 0x40, 0x0F, 0x80, // '@'
00039     0x18, 0x0C, 0x06, 0x04, 0x82, 0x43, 0xF1, 0x08, 0x84, 0x81, 0x00, // 'A'
00040     0xF9, 0x0A, 0x14, 0x2F, 0x90, 0xA1, 0x42, 0xF8, // 'B'
00041     0x3E, 0x40, 0x80, 0x80, 0x80, 0x80, 0x80, 0x40, 0x3C, // 'C'
00042     0xF8, 0x84, 0x82, 0x82, 0x82, 0x82, 0x84, 0xF8, // 'D'
00043     0xFA, 0x08, 0x20, 0xFA, 0x08, 0x20, 0xF8, // 'E'
00044     0xFA, 0x08, 0x20, 0xFA, 0x08, 0x20, 0x80, // 'F'
00045     0x3E, 0x40, 0x80, 0x80, 0x8E, 0x82, 0x82, 0x42, 0x3E, // 'G'
00046     0x82, 0x82, 0x82, 0x82, 0xFE, 0x82, 0x82, 0x82, // 'H'
00047     0xAA, 0xAA, 0x80, // 'I'

```



```

00048     0x22,0x22,0x22,0x22,0x22,0xC0, // 'J'
00049     0x89,0x12,0x45,0x0E,0x12,0x24,0x44,0x84, // 'K'
00050     0x82,0x08,0x20,0x82,0x08,0x20,0xF8, // 'L'
00051     0xC1,0xB0,0x6A,0x2A,0x8A,0xA2,0xA5,0x29,0x4A,0x22,0x88,0x80, // 'M'
00052     0xC2,0xC2,0xA2,0xA2,0x92,0x8A,0x8A,0x86,0x86, // 'N'
00053     0x78,0xC6,0x82,0x82,0x82,0x82,0x82,0xC6,0x78, // 'O'
00054     0xF2,0x28,0xA2,0x8B,0xC8,0x20,0x80, // 'P'
00055     0x78,0xC6,0x82,0x82,0x82,0x82,0x82,0xC6,0x7C,0x08,0x04, // 'Q'
00056     0xF1,0x12,0x24,0x4F,0x12,0x26,0x44,0x8C, // 'R'
00057     0x7A,0x08,0x30,0x70,0x20,0x82,0xF0, // 'S'
00058     0xFE,0x10,0x10,0x10,0x10,0x10,0x10,0x10, // 'T'
00059     0x82,0x82,0x82,0x82,0x82,0x82,0x82,0x44,0x78, // 'U'
00060     0x82,0x44,0x44,0x44,0x28,0x28,0x28,0x28,0x10, // 'V'
00061     0x84,0x24,0xC4,0x4A,0x44,0xA4,0x4A,0x43,0x18,0x31,0x83,0x18,0x31,0x80, // 'W'
00062     0x44,0x44,0x28,0x28,0x10,0x28,0x28,0x44,0x82, // 'X'
00063     0x82,0x44,0x44,0x28,0x38,0x10,0x10,0x10,0x10, // 'Y'
00064     0x7C,0x04,0x08,0x08,0x10,0x20,0x20,0x40,0xFE, // 'Z'
00065     0xE8,0x88,0x88,0x88,0xE0, // '['
00066     0x82,0x10,0x82,0x10,0x84,0x10, // '\'
00067     0xE2,0x22,0x22,0x22,0xE0, // ']'
00068     0x10,0x30,0x28,0x48,0x44,0x44, // '^'
00069     0xF8, // '_'
00070     0x44, // '`'
00071     0x70,0x27,0xA2,0x8B,0xE0, // 'a'
00072     0x82,0x08,0x3C,0x8A,0x28,0xA2,0xF0, // 'b'
00073     0x7A,0x08,0x20,0x81,0xC0, // 'c'
00074     0x08,0x20,0x9E,0x8A,0x28,0xA2,0x78, // 'd'
00075     0x72,0x2F,0xA0,0x81,0xE0, // 'e'
00076     0x31,0x04,0x3C,0x41,0x04,0x10,0x40, // 'f'
00077     0x3E,0x44,0x44,0x78,0x40,0x3C,0xC2,0x84,0x7C, // 'g'
00078     0x82,0x08,0x3E,0x8A,0x28,0xA2,0x88, // 'h'
00079     0x8A,0xAA, // 'i'
00080     0x20,0x22,0x22,0x22,0x22,0xE0, // 'j'
00081     0x82,0x08,0x24,0xA3,0x0A,0x24,0x90, // 'k'
00082     0xAA,0xAA,0x80, // 'l'
00083     0xF7,0x22,0x28,0x8A,0x22,0x88,0xA2,0x20, // 'm'
00084     0xFA,0x28,0xA2,0x8A,0x20, // 'n'
00085     0x72,0x28,0xA2,0x89,0xC0, // 'o'
00086     0xF2,0x28,0xA2,0x8B,0xC8,0x20,0x80, // 'p'
00087     0x7A,0x28,0xA2,0x89,0xE0,0x82,0x08, // 'q'
00088     0xB6,0x21,0x08,0x40, // 'r'
00089     0xF4,0x30,0x61,0x78, // 's'
00090     0x47,0x90,0x84,0x21,0xC0, // 't'
00091     0x8A,0x28,0xA2,0x8B,0xE0, // 'u'
00092     0x84,0x91,0x22,0x43,0x06,0x00, // 'v'
00093     0x88,0x95,0x45,0x51,0x54,0x65,0x08,0x80, // 'w'
00094     0x48,0x90,0xC1,0x84,0x99,0x00, // 'x'
00095     0x84,0x91,0x22,0x43,0x06,0x08,0x10,0xC0, // 'y'
00096     0xF0,0x42,0x10,0x83,0xE0, // 'z'
00097     0x18,0x82,0x08,0x23,0x02,0x08,0x20,0x81,0x80, // '{'
00098     0xAA,0xAA,0xAA, // '|'
00099     0xC0,0x82,0x08,0x20,0x62,0x08,0x20,0x8C,0x00 // '}'
00100 };
00101 const GFXglyph Open_Sans-Regular_12Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103 { 0, 2, 1, 4, 0, -1 }, // ' '
00104 { 1, 2, 9, 4, 1, -9 }, // '!'
00105 { 4, 4, 3, 6, 1, -9 }, // '"'
00106 { 6, 9, 9, 9, 0, -9 }, // '#'
00107 { 17, 6, 10, 8, 1, -9 }, // '$'
00108 { 25, 9, 9, 11, 1, -9 }, // '%'
00109 { 36, 9, 9, 10, 1, -9 }, // '&'
00110 { 47, 2, 3, 4, 1, -9 }, // "'"
00111 { 48, 4, 11, 5, 1, -9 }, // '('
00112 { 54, 4, 11, 5, 0, -9 }, // ')'
00113 { 60, 6, 5, 8, 1, -9 }, // '*'
00114 { 64, 6, 7, 8, 1, -8 }, // '+'
00115 { 70, 3, 3, 4, 0, -1 }, // ','
00116 { 72, 3, 1, 5, 1, -4 }, // '-'
00117 { 73, 2, 2, 4, 1, -2 }, // '.'
00118 { 74, 5, 9, 5, 0, -9 }, // '/'
00119 { 80, 6, 9, 8, 1, -9 }, // '0'
00120 { 87, 4, 9, 8, 1, -9 }, // '1'
00121 { 92, 6, 9, 8, 1, -9 }, // '2'
00122 { 99, 6, 9, 8, 1, -9 }, // '3'
00123 { 106, 8, 9, 8, 0, -9 }, // '4'
00124 { 115, 6, 9, 8, 1, -9 }, // '5'
00125 { 122, 6, 9, 8, 1, -9 }, // '6'
00126 { 129, 6, 9, 8, 1, -9 }, // '7'
00127 { 136, 6, 9, 8, 1, -9 }, // '8'
00128 { 143, 6, 9, 8, 1, -9 }, // '9'
00129 { 150, 2, 6, 4, 1, -6 }, // ':'
00130 { 152, 3, 8, 4, 0, -6 }, // ';'
00131 { 155, 6, 6, 8, 1, -7 }, // '<'
00132 { 160, 6, 3, 8, 1, -6 }, // '='
00133 { 163, 6, 6, 8, 1, -7 }, // '>'
00134 { 168, 6, 9, 6, 0, -9 }, // '?'

```

```

00135 { 175, 10, 10, 12, 1, -9 }, // '@'
00136 { 188, 9, 9, 9, 0, -9 }, // 'A'
00137 { 199, 7, 9, 9, 1, -9 }, // 'B'
00138 { 207, 8, 9, 9, 1, -9 }, // 'C'
00139 { 216, 8, 9, 10, 1, -9 }, // 'D'
00140 { 225, 6, 9, 8, 1, -9 }, // 'E'
00141 { 232, 6, 9, 7, 1, -9 }, // 'F'
00142 { 239, 8, 9, 10, 1, -9 }, // 'G'
00143 { 248, 8, 9, 10, 1, -9 }, // 'H'
00144 { 257, 2, 9, 4, 1, -9 }, // 'I'
00145 { 260, 4, 11, 4, -1, -9 }, // 'J'
00146 { 266, 7, 9, 8, 1, -9 }, // 'K'
00147 { 274, 6, 9, 7, 1, -9 }, // 'L'
00148 { 281, 10, 9, 12, 1, -9 }, // 'M'
00149 { 293, 8, 9, 10, 1, -9 }, // 'N'
00150 { 302, 8, 9, 10, 1, -9 }, // 'O'
00151 { 311, 6, 9, 8, 1, -9 }, // 'P'
00152 { 318, 8, 11, 10, 1, -9 }, // 'Q'
00153 { 329, 7, 9, 8, 1, -9 }, // 'R'
00154 { 337, 6, 9, 8, 1, -9 }, // 'S'
00155 { 344, 8, 9, 8, 0, -9 }, // 'T'
00156 { 353, 8, 9, 10, 1, -9 }, // 'U'
00157 { 362, 8, 9, 8, 0, -9 }, // 'V'
00158 { 371, 12, 9, 12, 0, -9 }, // 'W'
00159 { 385, 8, 9, 8, 0, -9 }, // 'X'
00160 { 394, 8, 9, 8, 0, -9 }, // 'Y'
00161 { 403, 8, 9, 8, 0, -9 }, // 'Z'
00162 { 412, 4, 11, 5, 1, -9 }, // '['
00163 { 418, 5, 9, 5, 0, -9 }, // '\'
00164 { 424, 4, 11, 5, 0, -9 }, // ']'
00165 { 430, 8, 6, 8, 0, -9 }, // '^'
00166 { 436, 6, 1, 6, 0, 1 }, // '_'
00167 { 437, 4, 2, 8, 2, -9 }, // '`'
00168 { 438, 6, 6, 8, 1, -6 }, // 'a'
00169 { 443, 6, 9, 8, 1, -9 }, // 'b'
00170 { 450, 6, 6, 7, 1, -6 }, // 'c'
00171 { 455, 6, 9, 8, 1, -9 }, // 'd'
00172 { 462, 6, 6, 8, 1, -6 }, // 'e'
00173 { 467, 6, 9, 5, 0, -9 }, // 'f'
00174 { 474, 8, 9, 8, 0, -6 }, // 'g'
00175 { 483, 6, 9, 8, 1, -9 }, // 'h'
00176 { 490, 2, 8, 4, 1, -8 }, // 'i'
00177 { 492, 4, 11, 4, -1, -8 }, // 'j'
00178 { 498, 6, 9, 7, 1, -9 }, // 'k'
00179 { 505, 2, 9, 4, 1, -9 }, // 'l'
00180 { 508, 10, 6, 12, 1, -6 }, // 'm'
00181 { 516, 6, 6, 8, 1, -6 }, // 'n'
00182 { 521, 6, 6, 8, 1, -6 }, // 'o'
00183 { 526, 6, 9, 8, 1, -6 }, // 'p'
00184 { 533, 6, 9, 8, 1, -6 }, // 'q'
00185 { 540, 5, 6, 6, 1, -6 }, // 'r'
00186 { 544, 5, 6, 7, 1, -6 }, // 's'
00187 { 548, 5, 7, 5, 0, -7 }, // 't'
00188 { 553, 6, 6, 8, 1, -6 }, // 'u'
00189 { 558, 7, 6, 7, 0, -6 }, // 'v'
00190 { 564, 10, 6, 10, 0, -6 }, // 'w'
00191 { 572, 7, 6, 7, 0, -6 }, // 'x'
00192 { 578, 7, 9, 7, 0, -6 }, // 'y'
00193 { 586, 6, 6, 7, 1, -6 }, // 'z'
00194 { 591, 6, 11, 6, 0, -9 }, // '{'
00195 { 600, 2, 12, 8, 3, -9 }, // '|'
00196 { 603, 6, 11, 6, 0, -9 }, // '}'
00197 };
00198 const GFXfont Open_Sans-Regular_12 PROGMEM = {
00199 (uint8_t *)Open_Sans-Regular_12Bitmaps, (GFXglyph *)Open_Sans-Regular_12Glyphs, 0x20, 0x7E, 17};

```

## 2.14 opensans9.h

```

00001 // Created by https://oledisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Open_Sans-Regular_9Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0x88, // '!'
00008     0xD8, // '"'
00009     0x28, 0x51, 0xF2, 0x8F, 0x8A, 0x14, 0x00, // '#'
00010     0x4E, 0xCC, 0x66, 0xE4, // '$'
00011     0x68, 0xA8, 0xB0, 0x7C, 0x1A, 0x2A, 0x2C, // '%'
00012     0x61, 0x22, 0x82, 0x0A, 0x53, 0x3E, 0x00, // '&'
00013     0xA0, // "'"
00014     0x44, 0x88, 0x88, 0x44, // '('
00015     0x44, 0x22, 0x22, 0x44, // ')'

```

```

00016     0x23, 0xE2, 0x14, // '*'
00017     0x20, 0x8F, 0x88, 0x20, // '+'
00018     0xA0, // ','
00019     0xE0, // '-'
00020     0x80, // '.'
00021     0x22, 0x44, 0x48, 0x80, // '/'
00022     0x71, 0x48, 0xA2, 0x89, 0x47, 0x00, // '0'
00023     0x59, 0x24, 0x90, // '1'
00024     0x70, 0x41, 0x04, 0x21, 0x0F, 0x80, // '2'
00025     0x70, 0x84, 0xC1, 0x0B, 0xC0, // '3'
00026     0x10, 0xC3, 0x14, 0x93, 0xE1, 0x00, // '4'
00027     0xE4, 0x21, 0xC1, 0x0B, 0x80, // '5'
00028     0x31, 0x04, 0x1C, 0x89, 0x27, 0x00, // '6'
00029     0xF8, 0x41, 0x04, 0x20, 0x84, 0x00, // '7'
00030     0x71, 0x45, 0x08, 0x52, 0x27, 0x00, // '8'
00031     0x72, 0x48, 0x9E, 0x08, 0x46, 0x00, // '9'
00032     0x80, 0x80, // ':'
00033     0x40, 0x09, 0x00, // ';'
00034     0x19, 0x86, 0x06, // '<'
00035     0xE0, 0xE0, // '='
00036     0xC0, 0xC3, 0x30, // '>'
00037     0xE0, 0x84, 0x44, 0x01, 0x00, // '?'
00038     0x79, 0x0A, 0xF6, 0x6C, 0xD7, 0xA0, 0x3C, // '@'
00039     0x30, 0x60, 0xC2, 0x47, 0x91, 0x21, 0x00, // 'A'
00040     0xF4, 0xA5, 0xC9, 0x4B, 0xC0, // 'B'
00041     0x7A, 0x08, 0x20, 0x82, 0x07, 0x00, // 'C'
00042     0xF2, 0x28, 0xA2, 0x8A, 0x2F, 0x00, // 'D'
00043     0xE8, 0x8E, 0x88, 0xE0, // 'E'
00044     0xF4, 0x21, 0xE8, 0x42, 0x00, // 'F'
00045     0x7A, 0x08, 0x26, 0x8A, 0x27, 0x80, // 'G'
00046     0x8A, 0x28, 0xBE, 0x8A, 0x28, 0x80, // 'H'
00047     0xAA, 0xA8, // 'I'
00048     0x49, 0x24, 0x92, 0xC0, // 'J'
00049     0x92, 0x4A, 0x30, 0xA2, 0x48, 0x80, // 'K'
00050     0x84, 0x21, 0x08, 0x43, 0xC0, // 'L'
00051     0x85, 0x9B, 0x36, 0x6B, 0x56, 0xA9, 0x00, // 'M'
00052     0x8B, 0x2C, 0xAA, 0x9A, 0x68, 0x80, // 'N'
00053     0x72, 0x28, 0xA2, 0x8A, 0x27, 0x00, // 'O'
00054     0xE4, 0xA5, 0x4E, 0x42, 0x00, // 'P'
00055     0x72, 0x28, 0xA2, 0x8A, 0x27, 0x04, // 'Q'
00056     0xF2, 0x49, 0x38, 0xA2, 0x48, 0x80, // 'R'
00057     0x72, 0x04, 0x0C, 0x08, 0x2F, 0x00, // 'S'
00058     0xF8, 0x82, 0x08, 0x20, 0x82, 0x00, // 'T'
00059     0x8A, 0x28, 0xA2, 0x8A, 0x27, 0x00, // 'U'
00060     0x8A, 0x25, 0x14, 0x50, 0x82, 0x00, // 'V'
00061     0x91, 0x4C, 0x96, 0x8B, 0x46, 0x63, 0x30, 0x90, // 'W'
00062     0x89, 0x45, 0x08, 0x51, 0x48, 0x80, // 'X'
00063     0x89, 0x45, 0x08, 0x20, 0x82, 0x00, // 'Y'
00064     0xF8, 0x41, 0x08, 0x41, 0x0F, 0x80, // 'Z'
00065     0xD2, 0x49, 0x26, // '['
00066     0x88, 0x44, 0x42, 0x20, // '\
00067     0xC9, 0x24, 0x96, // ']'
00068     0x21, 0x45, 0x22, // '^'
00069     0xF0, // '~'
00070     0xA0, // '̀'
00071     0x70, 0x9D, 0x27, 0x00, // 'a'
00072     0x84, 0x3D, 0x29, 0x4B, 0xC0, // 'b'
00073     0x72, 0x20, 0x87, 0x00, // 'c'
00074     0x10, 0xBD, 0x29, 0x4B, 0xC0, // 'd'
00075     0x71, 0x2F, 0x90, 0x70, // 'e'
00076     0x64, 0xE4, 0x44, 0x40, // 'f'
00077     0x7A, 0x47, 0x10, 0x72, 0x2F, 0x00, // 'g'
00078     0x84, 0x3D, 0x29, 0x4A, 0x40, // 'h'
00079     0x8A, 0xA8, // 'i'
00080     0x41, 0x24, 0x92, 0xC0, // 'j'
00081     0x84, 0x29, 0x8C, 0x52, 0x40, // 'k'
00082     0xAA, 0xA8, // 'l'
00083     0xED, 0x2A, 0x54, 0xA9, 0x40, // 'm'
00084     0xF4, 0xA5, 0x29, 0x00, // 'n'
00085     0x71, 0x48, 0x94, 0x70, // 'o'
00086     0xF4, 0xA5, 0x2F, 0x42, 0x00, // 'p'
00087     0xF4, 0xA5, 0x2F, 0x08, 0x40, // 'q'
00088     0xE8, 0x88, 0x80, // 'r'
00089     0x64, 0x18, 0x2E, 0x00, // 's'
00090     0x4E, 0x44, 0x46, // 't'
00091     0x94, 0xA5, 0x2F, 0x00, // 'u'
00092     0x89, 0x45, 0x14, 0x20, // 'v'
00093     0x92, 0xAA, 0x6C, 0x6C, 0x44, // 'w'
00094     0x51, 0x42, 0x14, 0x48, // 'x'
00095     0x89, 0x45, 0x14, 0x20, 0x8C, 0x00, // 'y'
00096     0xF1, 0x10, 0x8F, 0x00, // 'z'
00097     0x64, 0x44, 0x44, 0x46, // '{'
00098     0xAA, 0xAA, 0x80, // '|'
00099     0xC4, 0x44, 0x44, 0x4C // '}'
00100 };
00101 const GFXglyph Open_Sans-Regular_9Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset

```

```

00103 { 0, 2, 1, 3, 0, -1 }, // ' '
00104 { 1, 2, 7, 3, 1, -7 }, // '!'
00105 { 3, 3, 2, 5, 1, -7 }, // '"'
00106 { 4, 7, 7, 7, 0, -7 }, // '#'
00107 { 11, 4, 8, 6, 1, -7 }, // '$'
00108 { 15, 8, 7, 8, 0, -7 }, // '%'
00109 { 22, 7, 7, 8, 1, -7 }, // '&'
00110 { 29, 2, 2, 3, 1, -7 }, // "'"
00111 { 30, 4, 8, 4, 0, -7 }, // '('
00112 { 34, 4, 8, 4, 0, -7 }, // ')'
00113 { 38, 6, 4, 6, 0, -7 }, // '*'
00114 { 41, 6, 5, 6, 0, -6 }, // '+'
00115 { 45, 2, 2, 3, 0, -1 }, // ','
00116 { 46, 4, 1, 4, 0, -3 }, // '-'
00117 { 47, 2, 1, 3, 1, -1 }, // '.'
00118 { 48, 4, 7, 4, 0, -7 }, // '/'
00119 { 52, 6, 7, 6, 0, -7 }, // '0'
00120 { 58, 3, 7, 6, 1, -7 }, // '1'
00121 { 61, 6, 7, 6, 0, -7 }, // '2'
00122 { 67, 5, 7, 6, 0, -7 }, // '3'
00123 { 72, 6, 7, 6, 0, -7 }, // '4'
00124 { 78, 5, 7, 6, 1, -7 }, // '5'
00125 { 83, 6, 7, 6, 0, -7 }, // '6'
00126 { 89, 6, 7, 6, 0, -7 }, // '7'
00127 { 95, 6, 7, 6, 0, -7 }, // '8'
00128 { 101, 6, 7, 6, 0, -7 }, // '9'
00129 { 107, 2, 5, 3, 1, -5 }, // ':'
00130 { 109, 3, 6, 3, 0, -5 }, // ';'
00131 { 112, 6, 4, 6, 0, -5 }, // '<'
00132 { 115, 4, 3, 6, 1, -5 }, // '='
00133 { 117, 6, 4, 6, 0, -5 }, // '>'
00134 { 120, 5, 7, 5, 0, -7 }, // '?'
00135 { 125, 7, 8, 9, 1, -7 }, // '@'
00136 { 132, 7, 7, 7, 0, -7 }, // 'A'
00137 { 139, 5, 7, 7, 1, -7 }, // 'B'
00138 { 144, 6, 7, 7, 1, -7 }, // 'C'
00139 { 150, 6, 7, 8, 1, -7 }, // 'D'
00140 { 156, 4, 7, 6, 1, -7 }, // 'E'
00141 { 160, 5, 7, 6, 1, -7 }, // 'F'
00142 { 165, 6, 7, 8, 1, -7 }, // 'G'
00143 { 171, 6, 7, 8, 1, -7 }, // 'H'
00144 { 177, 2, 7, 4, 1, -7 }, // 'I'
00145 { 179, 3, 9, 3, -1, -7 }, // 'J'
00146 { 183, 6, 7, 7, 1, -7 }, // 'K'
00147 { 189, 5, 7, 6, 1, -7 }, // 'L'
00148 { 194, 7, 7, 9, 1, -7 }, // 'M'
00149 { 201, 6, 7, 8, 1, -7 }, // 'N'
00150 { 207, 6, 7, 8, 1, -7 }, // 'O'
00151 { 213, 5, 7, 6, 1, -7 }, // 'P'
00152 { 218, 6, 8, 8, 1, -7 }, // 'Q'
00153 { 224, 6, 7, 7, 1, -7 }, // 'R'
00154 { 230, 6, 7, 6, 0, -7 }, // 'S'
00155 { 236, 6, 7, 6, 0, -7 }, // 'T'
00156 { 242, 6, 7, 8, 1, -7 }, // 'U'
00157 { 248, 6, 7, 6, 0, -7 }, // 'V'
00158 { 254, 9, 7, 9, 0, -7 }, // 'W'
00159 { 262, 6, 7, 6, 0, -7 }, // 'X'
00160 { 268, 6, 7, 6, 0, -7 }, // 'Y'
00161 { 274, 6, 7, 6, 0, -7 }, // 'Z'
00162 { 280, 3, 8, 4, 1, -7 }, // '['
00163 { 283, 4, 7, 4, 0, -7 }, // '\'
00164 { 287, 3, 8, 4, 0, -7 }, // ']'
00165 { 290, 6, 4, 6, 0, -6 }, // '^'
00166 { 293, 5, 1, 5, 0, 1 }, // '_'
00167 { 294, 2, 2, 6, 2, -7 }, // '`'
00168 { 295, 5, 5, 6, 0, -5 }, // 'a'
00169 { 299, 5, 7, 7, 1, -7 }, // 'b'
00170 { 304, 5, 5, 5, 0, -5 }, // 'c'
00171 { 308, 5, 7, 7, 1, -7 }, // 'd'
00172 { 313, 6, 5, 6, 0, -5 }, // 'e'
00173 { 317, 4, 7, 4, 0, -7 }, // 'f'
00174 { 321, 6, 7, 6, 0, -5 }, // 'g'
00175 { 327, 5, 7, 7, 1, -7 }, // 'h'
00176 { 332, 2, 7, 3, 1, -7 }, // 'i'
00177 { 334, 3, 9, 3, -1, -7 }, // 'j'
00178 { 338, 5, 7, 6, 1, -7 }, // 'k'
00179 { 343, 2, 7, 3, 1, -7 }, // 'l'
00180 { 345, 7, 5, 9, 1, -5 }, // 'm'
00181 { 350, 5, 5, 7, 1, -5 }, // 'n'
00182 { 354, 6, 5, 6, 0, -5 }, // 'o'
00183 { 358, 5, 7, 7, 1, -5 }, // 'p'
00184 { 363, 5, 7, 7, 1, -5 }, // 'q'
00185 { 368, 4, 5, 5, 1, -5 }, // 'r'
00186 { 371, 5, 5, 5, 0, -5 }, // 's'
00187 { 375, 4, 6, 4, 0, -6 }, // 't'
00188 { 378, 5, 5, 7, 1, -5 }, // 'u'
00189 { 382, 6, 5, 6, 0, -5 }, // 'v'

```

```

00190     { 386, 8, 5, 8, 0, -5 }, // 'w'
00191     { 391, 6, 5, 6, 0, -5 }, // 'x'
00192     { 395, 6, 7, 6, 0, -5 }, // 'y'
00193     { 401, 5, 5, 5, 0, -5 }, // 'z'
00194     { 405, 4, 8, 4, 0, -7 }, // '{'
00195     { 409, 2, 9, 6, 2, -7 }, // '|'
00196     { 412, 4, 8, 4, 0, -7 } // '}'
00197 };
00198 const GFXfont Open_Sans-Regular_9 PROGMEM = {
00199 (uint8_t *)Open_Sans-Regular_9Bitmaps, (GFXglyph *)Open_Sans-Regular_9Glyphs, 0x20, 0x7E, 13};

```

## 2.15 roboto12.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Roboto_Condensed_Light_12Bitmaps[] PROGMEM = {
00004     // Bitmap Data:
00005     0x00, // ' '
00006     0xAA, 0xA8, 0x80, // '!'
00007     0xDB, 0x00, // '"'
00008     0x51, 0x4F, 0x94, 0x53, 0xEA, 0x28, 0xA0, // '#'
00009     0x43, 0x25, 0x28, 0x30, 0x52, 0x93, 0x90, // '$'
00010     0xC1, 0x52, 0xC6, 0x82, 0x05, 0x95, 0x2A, 0x0C, // '%'
00011     0x30, 0xA1, 0x43, 0x06, 0x12, 0xA6, 0x44, 0x74, // '&'
00012     0xA8, // '''
00013     0x4A, 0x49, 0x24, 0x92, 0x24, // '('
00014     0x84, 0x42, 0x22, 0x22, 0x22, 0x44, 0x80, // ')'
00015     0x22, 0xA7, 0x14, 0x90, // '*'
00016     0x20, 0x82, 0x3E, 0x20, 0x82, 0x00, // '+'
00017     0x4A, 0x00, // ','
00018     0xE0, // '-'
00019     0x80, // '.'
00020     0x11, 0x08, 0x44, 0x21, 0x08, 0x84, 0x00, // '/'
00021     0x64, 0xA5, 0x29, 0x4A, 0x52, 0x60, // '0'
00022     0x6A, 0x22, 0x22, 0x22, 0x20, // '1'
00023     0x64, 0xA4, 0x22, 0x11, 0x10, 0xF0, // '2'
00024     0x64, 0x84, 0x26, 0x08, 0x52, 0x60, // '3'
00025     0x10, 0x20, 0xC1, 0x85, 0x12, 0x3F, 0x08, 0x10, // '4'
00026     0x74, 0x21, 0xE9, 0x08, 0x52, 0x70, // '5'
00027     0x62, 0x21, 0xC9, 0x4A, 0x52, 0x60, // '6'
00028     0xF8, 0x20, 0x84, 0x10, 0x82, 0x08, 0x40, // '7'
00029     0x64, 0xA5, 0x26, 0x4A, 0x52, 0x70, // '8'
00030     0x64, 0xA5, 0x29, 0x38, 0x42, 0x60, // '9'
00031     0x80, 0x20, // ':'
00032     0x40, 0x00, 0x94, // ';'
00033     0x11, 0x91, 0x83, 0x08, // '<'
00034     0xF0, 0x3C, // '='
00035     0x82, 0x0C, 0x64, 0x40, // '>'
00036     0xEA, 0x22, 0x24, 0x40, 0x40, // '?'
00037     0x1E, 0x08, 0x44, 0xD2, 0x54, 0xA4, 0xA9, 0x4A, 0x52, 0x6C, 0x40, 0x18, 0x03, 0xC0, // '@'
00038     0x10, 0x30, 0x28, 0x28, 0x28, 0x48, 0x7C, 0x44, 0x86, // 'A'
00039     0xF2, 0x49, 0x24, 0xE2, 0x48, 0xA4, 0xF0, // 'B'
00040     0x72, 0x28, 0xA0, 0x82, 0x08, 0xA2, 0x70, // 'C'
00041     0xF2, 0x28, 0xA2, 0x8A, 0x28, 0xA2, 0xF0, // 'D'
00042     0xF4, 0x21, 0x0F, 0x42, 0x10, 0xF0, // 'E'
00043     0xF4, 0x21, 0x0F, 0x42, 0x10, 0x80, // 'F'
00044     0x72, 0x28, 0xA0, 0x82, 0x68, 0xA2, 0x78, // 'G'
00045     0x8A, 0x28, 0xA2, 0xFA, 0x28, 0xA2, 0x88, // 'H'
00046     0xAA, 0xAA, 0x80, // 'I'
00047     0x10, 0x84, 0x21, 0x0A, 0x52, 0xE0, // 'J'
00048     0x89, 0x22, 0x85, 0x0E, 0x14, 0x24, 0x44, 0x8C, // 'K'
00049     0x84, 0x21, 0x08, 0x42, 0x10, 0xF0, // 'L'
00050     0x82, 0x82, 0xC6, 0xC6, 0xCA, 0xAA, 0xAA, 0xB2, 0x92, // 'M'
00051     0x8A, 0x2C, 0xB2, 0xAA, 0xA9, 0xA2, 0x88, // 'N'
00052     0x72, 0x28, 0xA2, 0x8A, 0x28, 0xA2, 0x70, // 'O'
00053     0xF2, 0x48, 0xA2, 0x93, 0xC8, 0x20, 0x80, // 'P'
00054     0x72, 0x28, 0xA2, 0x8A, 0x28, 0xA2, 0x70, 0x20, // 'Q'
00055     0xF2, 0x28, 0xA2, 0x8B, 0xC9, 0x24, 0x88, // 'R'
00056     0x78, 0x92, 0x12, 0x03, 0x01, 0x21, 0x24, 0x78, // 'S'
00057     0xFC, 0x40, 0x81, 0x02, 0x04, 0x08, 0x10, 0x20, // 'T'
00058     0x8A, 0x28, 0xA2, 0x8A, 0x28, 0xA2, 0x70, // 'U'
00059     0x85, 0x09, 0x22, 0x44, 0x85, 0x0C, 0x18, 0x10, // 'V'
00060     0x88, 0xA2, 0x25, 0x51, 0x54, 0x55, 0x15, 0x46, 0x50, 0x88, 0x22, 0x00, // 'W'
00061     0xC4, 0x90, 0xA1, 0x81, 0x06, 0x12, 0x24, 0x84, // 'X'
00062     0x84, 0x91, 0x21, 0x83, 0x02, 0x08, 0x10, 0x20, // 'Y'
00063     0xF8, 0x41, 0x08, 0x41, 0x08, 0x20, 0xF8, // 'Z'
00064     0xD2, 0x49, 0x24, 0x92, 0x60, // '['
00065     0x84, 0x10, 0x84, 0x10, 0x84, 0x10, 0x80, // '\'
00066     0xC9, 0x24, 0x92, 0x49, 0x60, // ']'
00067     0x43, 0x29, 0x20, // '^'
00068     0xF8, // '_'
00069     0x88, // ``

```

```

00071      0x64, 0x84, 0xE9, 0x4B, 0xC0, // 'a'
00072      0x84, 0x3D, 0x29, 0x4A, 0x52, 0xF0, // 'b'
00073      0x72, 0x28, 0x20, 0x82, 0x27, 0x00, // 'c'
00074      0x10, 0xBD, 0x29, 0x4A, 0x52, 0xF0, // 'd'
00075      0x72, 0x48, 0xBE, 0x82, 0x07, 0x80, // 'e'
00076      0x68, 0xC8, 0x88, 0x88, 0x80, // 'f'
00077      0xF4, 0xA5, 0x29, 0x4B, 0xD2, 0x60, // 'g'
00078      0x84, 0x3D, 0x29, 0x4A, 0x52, 0x90, // 'h'
00079      0x8A, 0xAA, 0x80, // 'i'
00080      0x41, 0x24, 0x92, 0x4A, 0x00, // 'j'
00081      0x84, 0x2D, 0x4C, 0x62, 0x94, 0x90, // 'k'
00082      0xAA, 0xAA, 0x80, // 'l'
00083      0xEE, 0x92, 0x92, 0x92, 0x92, 0x92, // 'm'
00084      0xF4, 0xA5, 0x29, 0x4A, 0x40, // 'n'
00085      0x64, 0xA5, 0x29, 0x49, 0x80, // 'o'
00086      0xF4, 0xA5, 0x29, 0x4B, 0xD0, 0x80, // 'p'
00087      0xF4, 0xA5, 0x29, 0x4B, 0xC2, 0x10, // 'q'
00088      0xE8, 0x88, 0x88, 0x80, // 'r'
00089      0x72, 0x48, 0x18, 0x12, 0x67, 0x00, // 's'
00090      0x44, 0xE4, 0x44, 0x44, 0x60, // 't'
00091      0x94, 0xA5, 0x29, 0x49, 0xC0, // 'u'
00092      0x8A, 0x45, 0x14, 0x61, 0x82, 0x00, // 'v'
00093      0x92, 0x9A, 0x5A, 0x6A, 0x6C, 0x2C, 0x24, // 'w'
00094      0x99, 0x46, 0x08, 0x61, 0x48, 0x80, // 'x'
00095      0x8A, 0x45, 0x14, 0x60, 0x82, 0x10, 0x40, // 'y'
00096      0xF1, 0x08, 0x88, 0x43, 0xC0, // 'z'
00097      0x64, 0x88, 0x88, 0x88, 0x84, 0x60, // '{'
00098      0xAA, 0xAA, 0xA8, // '|'
00099      0xC4, 0x44, 0x42, 0x44, 0x44, 0xC0 // '}'
00100  };
00101  const GFXglyph Roboto_Condensed_Light_12Glyphs[] PROGMEM = {
00102  // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103      { 0, 2, 1, 4, 0, -1 }, // ' '
00104      { 1, 2, 9, 4, 1, -9 }, // '!'
00105      { 4, 3, 3, 4, 1, -9 }, // '"'
00106      { 6, 6, 9, 7, 1, -9 }, // '#'
00107      { 13, 5, 11, 7, 1, -10 }, // '$'
00108      { 20, 7, 9, 9, 1, -9 }, // '%'
00109      { 28, 7, 9, 7, 0, -9 }, // '&'
00110      { 36, 2, 3, 3, 1, -9 }, // "'"
00111      { 37, 3, 13, 5, 1, -10 }, // '('
00112      { 42, 4, 13, 5, 0, -10 }, // ')'
00113      { 49, 6, 5, 6, 0, -9 }, // '*'
00114      { 53, 6, 7, 7, 0, -7 }, // '+'
00115      { 59, 3, 3, 3, 0, -1 }, // ','
00116      { 61, 4, 1, 4, 0, -4 }, // '-'
00117      { 62, 2, 1, 4, 1, -1 }, // '.'
00118      { 63, 5, 10, 5, 0, -9 }, // '/'
00119      { 70, 5, 9, 7, 1, -9 }, // '0'
00120      { 76, 4, 9, 7, 1, -9 }, // '1'
00121      { 81, 5, 9, 7, 1, -9 }, // '2'
00122      { 87, 5, 9, 7, 1, -9 }, // '3'
00123      { 93, 7, 9, 7, 0, -9 }, // '4'
00124      { 101, 5, 9, 7, 1, -9 }, // '5'
00125      { 107, 5, 9, 7, 1, -9 }, // '6'
00126      { 113, 6, 9, 7, 0, -9 }, // '7'
00127      { 120, 5, 9, 7, 1, -9 }, // '8'
00128      { 126, 5, 9, 7, 1, -9 }, // '9'
00129      { 132, 2, 6, 3, 1, -6 }, // ':'
00130      { 134, 3, 8, 3, 0, -6 }, // ';'
00131      { 137, 5, 6, 6, 0, -7 }, // '<'
00132      { 141, 5, 3, 7, 1, -5 }, // '='
00133      { 143, 5, 6, 6, 1, -7 }, // '>'
00134      { 147, 4, 9, 6, 1, -9 }, // '?'
00135      { 152, 10, 11, 10, 0, -8 }, // '@'
00136      { 166, 8, 9, 8, 0, -9 }, // 'A'
00137      { 175, 6, 9, 7, 1, -9 }, // 'B'
00138      { 182, 6, 9, 8, 1, -9 }, // 'C'
00139      { 189, 6, 9, 8, 1, -9 }, // 'D'
00140      { 196, 5, 9, 7, 1, -9 }, // 'E'
00141      { 202, 5, 9, 7, 1, -9 }, // 'F'
00142      { 208, 6, 9, 8, 1, -9 }, // 'G'
00143      { 215, 6, 9, 8, 1, -9 }, // 'H'
00144      { 222, 2, 9, 4, 1, -9 }, // 'I'
00145      { 225, 5, 9, 7, 1, -9 }, // 'J'
00146      { 231, 7, 9, 8, 1, -9 }, // 'K'
00147      { 239, 5, 9, 7, 1, -9 }, // 'L'
00148      { 245, 8, 9, 10, 1, -9 }, // 'M'
00149      { 254, 6, 9, 8, 1, -9 }, // 'N'
00150      { 261, 6, 9, 8, 1, -9 }, // 'O'
00151      { 268, 6, 9, 7, 1, -9 }, // 'P'
00152      { 275, 6, 10, 8, 1, -9 }, // 'Q'
00153      { 283, 6, 9, 8, 1, -9 }, // 'R'
00154      { 290, 7, 9, 7, 0, -9 }, // 'S'
00155      { 298, 7, 9, 7, 0, -9 }, // 'T'
00156      { 306, 6, 9, 8, 1, -9 }, // 'U'
00157      { 313, 7, 9, 7, 0, -9 }, // 'V'

```

```

00158     { 321, 10, 9, 10, 0, -9 }, // 'W'
00159     { 333, 7, 9, 7, 0, -9 }, // 'X'
00160     { 341, 7, 9, 7, 0, -9 }, // 'Y'
00161     { 349, 6, 9, 7, 1, -9 }, // 'Z'
00162     { 356, 3, 12, 4, 1, -10 }, // '['
00163     { 361, 5, 10, 5, 0, -9 }, // '\'
00164     { 368, 3, 12, 4, 0, -10 }, // ']'
00165     { 373, 5, 4, 5, 0, -9 }, // '^'
00166     { 376, 6, 1, 6, 0, 0 }, // '_'
00167     { 377, 3, 2, 4, 1, -9 }, // '`'
00168     { 378, 5, 7, 7, 1, -7 }, // 'a'
00169     { 383, 5, 9, 7, 1, -9 }, // 'b'
00170     { 389, 6, 7, 6, 0, -7 }, // 'c'
00171     { 395, 5, 9, 7, 1, -9 }, // 'd'
00172     { 401, 6, 7, 6, 0, -7 }, // 'e'
00173     { 407, 4, 9, 5, 1, -9 }, // 'f'
00174     { 412, 5, 9, 7, 1, -7 }, // 'g'
00175     { 418, 5, 9, 7, 1, -9 }, // 'h'
00176     { 424, 2, 9, 4, 1, -9 }, // 'i'
00177     { 427, 3, 11, 4, 0, -9 }, // 'j'
00178     { 432, 5, 9, 6, 1, -9 }, // 'k'
00179     { 438, 2, 9, 4, 1, -9 }, // 'l'
00180     { 441, 8, 7, 10, 1, -7 }, // 'm'
00181     { 448, 5, 7, 7, 1, -7 }, // 'n'
00182     { 453, 5, 7, 7, 1, -7 }, // 'o'
00183     { 458, 5, 9, 7, 1, -7 }, // 'p'
00184     { 464, 5, 9, 7, 1, -7 }, // 'q'
00185     { 470, 4, 7, 5, 1, -7 }, // 'r'
00186     { 474, 6, 7, 6, 0, -7 }, // 's'
00187     { 480, 4, 9, 4, 0, -9 }, // 't'
00188     { 485, 5, 7, 7, 1, -7 }, // 'u'
00189     { 490, 6, 7, 6, 0, -7 }, // 'v'
00190     { 496, 8, 7, 9, 0, -7 }, // 'w'
00191     { 503, 6, 7, 6, 0, -7 }, // 'x'
00192     { 509, 6, 9, 6, 0, -7 }, // 'y'
00193     { 516, 5, 7, 6, 1, -7 }, // 'z'
00194     { 521, 4, 11, 5, 1, -9 }, // '{'
00195     { 527, 2, 11, 4, 1, -9 }, // '|'
00196     { 530, 4, 11, 5, 0, -9 }, // '}'
00197 };
00198 const GFXfont RobotoCondensedLight12 PROGMEM = {
00199 (uint8_t *)RobotoCondensedLight12Bitmaps, (GFXglyph *)RobotoCondensedLight12Glyphs, 0x20, 0x7E,
00200 15};

```

## 2.16 roboto9.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t RobotoCondensedLight9Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0x88, // '!'
00008     0xA0, // '"'
00009     0x51, 0x47, 0x98, 0xF1, 0x86, 0x00, // '#'
00010     0x64, 0xA4, 0xC1, 0x49, 0x88, // '$'
00011     0xC7, 0x38, 0x87, 0x58, 0xC0, // '%'
00012     0xC6, 0x31, 0x0E, 0x53, 0xC0, // '&'
00013     0xA0, // "'"
00014     0xD2, 0x49, 0x24, 0xC0, // '('
00015     0x89, 0x24, 0x92, 0x80, // ')'
00016     0x47, 0xA8, // '*'
00017     0x42, 0x3C, 0x84, 0x00, // '+'
00018     0xA0, // ','
00019     0xC0, // '-'
00020     0x80, // '.'
00021     0x24, 0x44, 0x48, 0x88, // '/'
00022     0x64, 0xA5, 0x29, 0x49, 0x80, // '0'
00023     0x59, 0x24, 0x90, // '1'
00024     0x64, 0x88, 0x44, 0x43, 0xC0, // '2'
00025     0x64, 0x88, 0xC1, 0x49, 0x80, // '3'
00026     0x21, 0x18, 0xCA, 0x78, 0x80, // '4'
00027     0x74, 0x19, 0x21, 0x49, 0x80, // '5'
00028     0x64, 0x39, 0x29, 0x49, 0x80, // '6'
00029     0xF0, 0x88, 0x42, 0x21, 0x00, // '7'
00030     0x64, 0xA8, 0xC9, 0x49, 0x80, // '8'
00031     0x65, 0x25, 0x27, 0x11, 0x80, // '9'
00032     0x80, 0x80, // ':'
00033     0x80, 0xA0, // ';'
00034     0x24, 0xC2, // '<'
00035     0xE0, 0xE0, // '='
00036     0x84, 0x68, // '>'
00037     0xD9, 0x28, 0x20, // '?'

```

```

00038      0x38, 0x44, 0x9C, 0xAA, 0xAC, 0xBC, 0x40, 0x38, // '@'
00039      0x21, 0x86, 0x14, 0x51, 0xC8, 0x80, // 'A'
00040      0xEA, 0xAE, 0xAA, 0xE0, // 'B'
00041      0xE5, 0x25, 0x08, 0x5B, 0x80, // 'C'
00042      0xEA, 0xAA, 0xAA, 0xE0, // 'D'
00043      0xE8, 0x8E, 0x88, 0xE0, // 'E'
00044      0xE8, 0x8E, 0x88, 0x80, // 'F'
00045      0x74, 0xA1, 0x09, 0x49, 0xC0, // 'G'
00046      0x94, 0xA5, 0xE9, 0x4A, 0x40, // 'H'
00047      0xAA, 0xA8, // 'I'
00048      0x22, 0x22, 0x2A, 0x60, // 'J'
00049      0xB5, 0x31, 0x8C, 0x52, 0xC0, // 'K'
00050      0x88, 0x88, 0x88, 0xE0, // 'L'
00051      0x8A, 0x68, 0xB6, 0xEB, 0xAA, 0x80, // 'M'
00052      0x94, 0xB5, 0xAB, 0x5A, 0x40, // 'N'
00053      0x72, 0x48, 0xA2, 0x8A, 0x47, 0x00, // 'O'
00054      0xE5, 0x25, 0x4E, 0x42, 0x00, // 'P'
00055      0x72, 0x48, 0xA2, 0x8A, 0x47, 0x06, // 'Q'
00056      0xE5, 0x29, 0x4E, 0x52, 0xC0, // 'R'
00057      0xEA, 0x84, 0x2A, 0xE0, // 'S'
00058      0xF1, 0x08, 0x42, 0x10, 0x80, // 'T'
00059      0xAA, 0xAA, 0xAA, 0xE0, // 'U'
00060      0x8A, 0x49, 0x14, 0x61, 0x82, 0x00, // 'V'
00061      0x92, 0xB4, 0xB4, 0xB4, 0x6C, 0x4C, 0x48, // 'W'
00062      0x91, 0x46, 0x08, 0x61, 0x49, 0x80, // 'X'
00063      0x9A, 0x46, 0x08, 0x20, 0x82, 0x00, // 'Y'
00064      0xF0, 0x88, 0x44, 0x43, 0xC0, // 'Z'
00065      0xAA, 0xAA, // '['
00066      0x88, 0x84, 0x44, 0x22, // '\ '
00067      0xAA, 0xAA, // ']'
00068      0x4C, 0xA0, // '^'
00069      0xE0, // '_'
00070      0x88, // '`'
00071      0x65, 0x19, 0x47, 0x00, // 'a'
00072      0x84, 0x39, 0x29, 0x4B, 0x80, // 'b'
00073      0x64, 0xA1, 0x26, 0x00, // 'c'
00074      0x10, 0x9D, 0x29, 0x49, 0xC0, // 'd'
00075      0x64, 0xBD, 0x07, 0x00, // 'e'
00076      0x64, 0xE4, 0x44, 0x40, // 'f'
00077      0x74, 0xA5, 0x27, 0x49, 0x80, // 'g'
00078      0x84, 0x39, 0x49, 0x4A, 0x40, // 'h'
00079      0x8A, 0xA8, // 'i'
00080      0x8A, 0xAA, 0x80, // 'j'
00081      0x88, 0xE8, 0x8C, 0xE0, // 'k'
00082      0xAA, 0xA8, // 'l'
00083      0xDA, 0xAA, 0xAA, 0xA8, // 'm'
00084      0xE5, 0x25, 0x29, 0x00, // 'n'
00085      0x64, 0xA5, 0x26, 0x00, // 'o'
00086      0xE4, 0xA5, 0x2E, 0x42, 0x00, // 'p'
00087      0x74, 0xA5, 0x27, 0x08, 0x40, // 'q'
00088      0xD2, 0x48, // 'r'
00089      0x64, 0x18, 0x4F, 0x00, // 's'
00090      0x59, 0x24, 0x80, // 't'
00091      0x94, 0xA5, 0x27, 0x00, // 'u'
00092      0x95, 0x18, 0xC4, 0x00, // 'v'
00093      0xAD, 0x72, 0xE2, 0x45, 0x00, // 'w'
00094      0xB3, 0x10, 0xCB, 0x00, // 'x'
00095      0x95, 0x18, 0xC4, 0x21, 0x00, // 'y'
00096      0xE1, 0x11, 0x0F, 0x00, // 'z'
00097      0x64, 0x44, 0x84, 0x44, 0x60, // '{'
00098      0xAA, 0xAA, // '|'
00099      0x84, 0x44, 0x24, 0x44, 0x80 // '}'
00100  };
00101  const GFXglyph Roboto_Condensed_Light_9Glyphs[] PROGMEM = {
00102  // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103      { 0, 2, 1, 3, 0, -1 }, // ' '
00104      { 1, 2, 7, 3, 1, -7 }, // '!'
00105      { 3, 2, 2, 4, 1, -7 }, // '"'
00106      { 4, 6, 7, 6, 0, -7 }, // '#'
00107      { 10, 5, 8, 5, 0, -7 }, // '$'
00108      { 15, 5, 7, 7, 1, -7 }, // '%'
00109      { 20, 5, 7, 6, 1, -7 }, // '&'
00110      { 25, 2, 2, 3, 0, -7 }, // "'"
00111      { 26, 3, 9, 4, 1, -7 }, // '('
00112      { 30, 3, 9, 4, 0, -7 }, // ')'
00113      { 34, 5, 3, 5, 0, -7 }, // '*'
00114      { 36, 5, 5, 5, 0, -5 }, // '+'
00115      { 40, 2, 2, 3, 0, -1 }, // ','
00116      { 41, 3, 1, 3, 0, -3 }, // '-'
00117      { 42, 2, 1, 3, 0, -1 }, // '.'
00118      { 43, 4, 8, 4, 0, -7 }, // '/'
00119      { 47, 5, 7, 5, 0, -7 }, // '0'
00120      { 52, 3, 7, 5, 1, -7 }, // '1'
00121      { 55, 5, 7, 5, 0, -7 }, // '2'
00122      { 60, 5, 7, 5, 0, -7 }, // '3'
00123      { 65, 5, 7, 5, 0, -7 }, // '4'
00124      { 70, 5, 7, 5, 0, -7 }, // '5'

```



```

00125 { 75, 5, 7, 5, 0, -7 }, // '6'
00126 { 80, 5, 7, 5, 0, -7 }, // '7'
00127 { 85, 5, 7, 5, 0, -7 }, // '8'
00128 { 90, 5, 7, 5, 0, -7 }, // '9'
00129 { 95, 2, 5, 3, 0, -5 }, // ':'
00130 { 97, 2, 6, 3, 0, -5 }, // ';'
00131 { 99, 4, 4, 5, 0, -5 }, // '<'
00132 { 101, 4, 3, 5, 1, -4 }, // '='
00133 { 103, 4, 4, 5, 1, -5 }, // '>'
00134 { 105, 3, 7, 5, 1, -7 }, // '?'
00135 { 108, 8, 8, 8, 0, -6 }, // '@'
00136 { 116, 6, 7, 6, 0, -7 }, // 'A'
00137 { 122, 4, 7, 6, 1, -7 }, // 'B'
00138 { 126, 5, 7, 6, 1, -7 }, // 'C'
00139 { 131, 4, 7, 6, 1, -7 }, // 'D'
00140 { 135, 4, 7, 5, 1, -7 }, // 'E'
00141 { 139, 4, 7, 5, 1, -7 }, // 'F'
00142 { 143, 5, 7, 6, 0, -7 }, // 'G'
00143 { 148, 5, 7, 7, 1, -7 }, // 'H'
00144 { 153, 2, 7, 3, 1, -7 }, // 'I'
00145 { 155, 4, 7, 5, 0, -7 }, // 'J'
00146 { 159, 5, 7, 6, 1, -7 }, // 'K'
00147 { 164, 4, 7, 5, 1, -7 }, // 'L'
00148 { 168, 6, 7, 8, 1, -7 }, // 'M'
00149 { 174, 5, 7, 7, 1, -7 }, // 'N'
00150 { 179, 6, 7, 6, 0, -7 }, // 'O'
00151 { 185, 5, 7, 6, 1, -7 }, // 'P'
00152 { 190, 6, 8, 6, 0, -7 }, // 'Q'
00153 { 196, 5, 7, 6, 1, -7 }, // 'R'
00154 { 201, 4, 7, 6, 1, -7 }, // 'S'
00155 { 205, 5, 7, 6, 0, -7 }, // 'T'
00156 { 210, 4, 7, 6, 1, -7 }, // 'U'
00157 { 214, 6, 7, 6, 0, -7 }, // 'V'
00158 { 220, 8, 7, 8, 0, -7 }, // 'W'
00159 { 227, 6, 7, 6, 0, -7 }, // 'X'
00160 { 233, 6, 7, 6, 0, -7 }, // 'Y'
00161 { 239, 5, 7, 6, 0, -7 }, // 'Z'
00162 { 244, 2, 8, 3, 1, -7 }, // '['
00163 { 246, 4, 8, 4, 0, -7 }, // '\'
00164 { 250, 2, 8, 3, 0, -7 }, // ']'
00165 { 252, 4, 3, 4, 0, -7 }, // '^'
00166 { 254, 4, 1, 4, 0, 0 }, // '_'
00167 { 255, 3, 2, 4, 0, -8 }, // '`'
00168 { 256, 5, 5, 5, 0, -5 }, // 'a'
00169 { 260, 5, 7, 5, 0, -7 }, // 'b'
00170 { 265, 5, 5, 5, 0, -5 }, // 'c'
00171 { 269, 5, 7, 5, 0, -7 }, // 'd'
00172 { 274, 5, 5, 5, 0, -5 }, // 'e'
00173 { 278, 4, 7, 4, 0, -7 }, // 'f'
00174 { 282, 5, 7, 5, 0, -5 }, // 'g'
00175 { 287, 5, 7, 5, 0, -7 }, // 'h'
00176 { 292, 2, 7, 3, 1, -7 }, // 'i'
00177 { 294, 2, 9, 3, 0, -7 }, // 'j'
00178 { 297, 4, 7, 5, 1, -7 }, // 'k'
00179 { 301, 2, 7, 3, 0, -7 }, // 'l'
00180 { 303, 6, 5, 8, 1, -5 }, // 'm'
00181 { 307, 5, 5, 5, 0, -5 }, // 'n'
00182 { 311, 5, 5, 5, 0, -5 }, // 'o'
00183 { 315, 5, 7, 5, 0, -5 }, // 'p'
00184 { 320, 5, 7, 5, 0, -5 }, // 'q'
00185 { 325, 3, 5, 4, 1, -5 }, // 'r'
00186 { 327, 5, 5, 5, 0, -5 }, // 's'
00187 { 331, 3, 6, 4, 0, -6 }, // 't'
00188 { 334, 5, 5, 5, 0, -5 }, // 'u'
00189 { 338, 5, 5, 5, 0, -5 }, // 'v'
00190 { 342, 7, 5, 7, 0, -5 }, // 'w'
00191 { 347, 5, 5, 5, 0, -5 }, // 'x'
00192 { 351, 5, 7, 5, 0, -5 }, // 'y'
00193 { 356, 5, 5, 5, 0, -5 }, // 'z'
00194 { 360, 4, 9, 4, 0, -7 }, // '{'
00195 { 365, 2, 8, 3, 1, -7 }, // '|'
00196 { 367, 4, 9, 4, 0, -7 }, // '}'
00197 };
00198 const GFXfont RobotoCondensed_Light_9 PROGMEM = {
00199 (uint8_t *)RobotoCondensed_Light_9Bitmaps, (GFXglyph *)RobotoCondensed_Light_9Glyphs, 0x20, 0x7E,
00200 12};

```

## 2.17 SansSerif12.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t SansSerif_plain_12Bitmaps[] PROGMEM = {
00004

```

```

00005 // Bitmap Data:
00006 0x00, // ' '
00007 0xAA,0xA2,0x80, // '!'
00008 0xAA,0xA0, // '"'
00009 0x12,0x0A,0x1F,0xC4,0x82,0x47,0xF0,0xA0,0x90, // '#'
00010 0x21,0xCA,0xA8,0xE0,0xE2,0xAA,0x70,0x82,0x00, // '$'
00011 0x61,0x12,0x42,0x48,0x4A,0x06,0xD8,0x14,0x84,0x90,0x92,0x21,0x80, // '%'
00012 0x30,0x24,0x10,0x0C,0x05,0x14,0x4A,0x19,0x88,0x7B,0x00, // '&'
00013 0xA8, // '''
00014 0x64,0x48,0x88,0x88,0x44,0x60, // '('
00015 0xC4,0x42,0x22,0x22,0x44,0xC0, // ')'
00016 0x22,0xA7,0x1C,0xA8,0x80, // '*'
00017 0x10,0x10,0x10,0xFE,0x10,0x10,0x10, // '+'
00018 0xA8, // ','
00019 0xE0, // '-'
00020 0xA0, // '.'
00021 0x10,0x88,0x42,0x21,0x08,0x84,0x00, // '/'
00022 0x78,0x92,0x14,0x28,0x50,0xA1,0x24,0x78, // '0'
00023 0xE0,0x82,0x08,0x20,0x82,0x08,0xF8, // '1'
00024 0x79,0x18,0x10,0x20,0x82,0x08,0x20,0xFC, // '2'
00025 0x79,0x08,0x10,0x23,0x80,0x81,0x42,0x78, // '3'
00026 0x18,0x30,0xA2,0x44,0x91,0x3F,0x04,0x08, // '4'
00027 0xF9,0x02,0x07,0xC0,0xC0,0x81,0x46,0x78, // '5'
00028 0x38,0x8A,0x05,0xCC,0xD0,0xA1,0x26,0x78, // '6'
00029 0xFC,0x08,0x20,0x41,0x02,0x08,0x10,0x40, // '7'
00030 0x79,0x0A,0x14,0x27,0x90,0xA1,0x42,0x78, // '8'
00031 0x79,0x92,0x14,0x2C,0xCE,0x81,0x44,0x70, // '9'
00032 0xA0,0xA0, // ':'
00033 0xA0,0xA8, // ';'
00034 0x03,0x0F,0x38,0x1C,0x01,0xE0,0x18, // '<'
00035 0xFF,0x00,0x3F,0xC0, // '='
00036 0xC0,0x3C,0x01,0xC0,0xE7,0x86,0x00, // '>'
00037 0x72,0x20,0x84,0x20,0x80,0x08,0x20, // '?'
00038 0x1F,0x02,0x0C,0x40,0x48,0xF2,0x91,0x29,0x12,0x91,0x48,0xF8,0x40,0x02,0x08,0x1F,0x00, // '@'
00039 0x18,0x0C,0x09,0x04,0x82,0x42,0x11,0xF8,0x84,0x81,0x00, // 'A'
00040 0xF9,0x0A,0x14,0x2F,0x90,0xA1,0x42,0xF8, // 'B'
00041 0x38,0x8A,0x04,0x08,0x10,0x20,0x22,0x38, // 'C'
00042 0xF8,0x84,0x82,0x82,0x82,0x82,0x82,0x84,0xF8, // 'D'
00043 0xFD,0x02,0x04,0x0F,0xD0,0x20,0x40,0xFC, // 'E'
00044 0xFA,0x08,0x20,0xFA,0x08,0x20,0x80, // 'F'
00045 0x3C,0x42,0x80,0x80,0x8E,0x82,0x82,0x42,0x3C, // 'G'
00046 0x82,0x82,0x82,0x82,0xFE,0x82,0x82,0x82,0x82, // 'H'
00047 0xAA,0xAA,0x80, // 'I'
00048 0x22,0x22,0x22,0x22,0x22,0xC0, // 'J'
00049 0x84,0x88,0x90,0xA0,0xC0,0xA0,0x90,0x88,0x84, // 'K'
00050 0x82,0x08,0x20,0x82,0x08,0x20,0xF8, // 'L'
00051 0x81,0x61,0xB0,0xD4,0xAA,0x54,0xCA,0x65,0x02,0x81,0x00, // 'M'
00052 0xC2,0xC2,0xA2,0xA2,0x92,0x8A,0x8A,0x86,0x86, // 'N'
00053 0x38,0x44,0x82,0x82,0x82,0x82,0x44,0x38, // 'O'
00054 0xF9,0x0A,0x14,0x2F,0x90,0x20,0x40,0x80, // 'P'
00055 0x38,0x44,0x82,0x82,0x82,0x82,0x44,0x38,0x08,0x04, // 'Q'
00056 0xF8,0x84,0x84,0x84,0xF8,0x88,0x84,0x84,0x82, // 'R'
00057 0x79,0x0A,0x04,0x07,0x80,0x81,0x42,0x78, // 'S'
00058 0xFE,0x10,0x10,0x10,0x10,0x10,0x10,0x10, // 'T'
00059 0x82,0x82,0x82,0x82,0x82,0x82,0x82,0xC6,0x7C, // 'U'
00060 0x40,0x88,0x10,0x84,0x10,0x82,0x10,0x24,0x04,0x80,0x60,0x0C,0x00, // 'V'
00061 0x84,0x24,0x44,0x44,0x44,0xA4,0x2A,0x82,0xA8,0x2A,0x81,0x10,0x11,0x00, // 'W'
00062 0xC6,0x44,0x28,0x28,0x10,0x28,0x28,0x44,0x82, // 'X'
00063 0x82,0x44,0x44,0x28,0x28,0x10,0x10,0x10,0x10, // 'Y'
00064 0xFE,0x02,0x04,0x08,0x10,0x20,0x40,0x80,0xFE, // 'Z'
00065 0xD2,0x49,0x24,0x93,0x00, // '['
00066 0x84,0x10,0x84,0x10,0x84,0x10,0x80, // '\'
00067 0xC9,0x24,0x92,0x4B,0x00, // ']'
00068 0x18,0x12,0x10,0x80, // '^'
00069 0xFC, // '_'
00070 0x42, // '`'
00071 0x79,0x08,0x13,0xE8,0x51,0x9D,0x00, // 'a'
00072 0x81,0x02,0x07,0xCC,0xD0,0xA1,0x42,0xCD,0xF0, // 'b'
00073 0x73,0x28,0x20,0x83,0x27,0x00, // 'c'
00074 0x04,0x08,0x13,0xEC,0xD0,0xA1,0x42,0xCC,0xF8, // 'd'
00075 0x79,0x9A,0x17,0xE8,0x18,0x9E,0x00, // 'e'
00076 0x32,0x11,0xE4,0x21,0x08,0x42,0x00, // 'f'
00077 0x7D,0x9A,0x14,0x28,0x59,0x9F,0x02,0x4C,0x70, // 'g'
00078 0x81,0x02,0x05,0xCC,0x50,0xA1,0x42,0x85,0x08, // 'h'
00079 0x8A,0xAA,0x80, // 'i'
00080 0x41,0x24,0x92,0x49,0x60, // 'j'
00081 0x81,0x02,0x04,0x49,0x14,0x30,0x50,0x91,0x10, // 'k'
00082 0xAA,0xAA,0xA0, // 'l'
00083 0xF7,0x22,0x28,0x8A,0x22,0x88,0xA2,0x28,0x88, // 'm'
00084 0xB9,0x8A,0x14,0x28,0x50,0xA1,0x00, // 'n'
00085 0x79,0x9A,0x14,0x28,0x59,0x9E,0x00, // 'o'
00086 0xF9,0x9A,0x14,0x28,0x59,0xBE,0x40,0x81,0x00, // 'p'
00087 0x7D,0x9A,0x14,0x28,0x59,0x9F,0x02,0x04,0x08, // 'q'
00088 0xB6,0x21,0x08,0x42,0x00, // 'r'
00089 0x72,0x28,0x1C,0x0A,0x27,0x00, // 's'
00090 0x42,0x3C,0x84,0x21,0x08,0x70, // 't'
00091 0x85,0x0A,0x14,0x28,0x51,0x9D,0x00, // 'u'

```

```

00092     0x85,0x09,0x22,0x44,0x86,0x0C,0x00, // 'v'
00093     0x88,0xA2,0x25,0x51,0x54,0x55,0x08,0x82,0x20, // 'w'
00094     0x84,0x91,0x21,0x84,0x89,0x21,0x00, // 'x'
00095     0x85,0x09,0x22,0x42,0x86,0x04,0x08,0x21,0x80, // 'y'
00096     0xF8,0x21,0x08,0x42,0x0F,0x80, // 'z'
00097     0x38,0x82,0x08,0x23,0x02,0x08,0x20,0x83,0x80, // '{'
00098     0xAA,0xAA,0xAA, // '|'
00099     0xE0,0x82,0x08,0x20,0x62,0x08,0x20,0x8E,0x00 // '}'
00100 };
00101 const GFXglyph SansSerif_plain_l2Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103     { 0, 2, 1, 5, 0, -1 }, // ' '
00104     { 1, 2, 9, 6, 2, -9 }, // '!'
00105     { 4, 4, 3, 6, 1, -9 }, // '"'
00106     { 6, 9, 8, 11, 1, -8 }, // '#'
00107     { 15, 6, 11, 9, 2, -9 }, // '$'
00108     { 24, 11, 9, 12, 0, -9 }, // '%'
00109     { 37, 9, 9, 11, 1, -9 }, // '&'
00110     { 48, 2, 3, 4, 1, -9 }, // "'"
00111     { 49, 4, 11, 6, 1, -10 }, // '('
00112     { 55, 4, 11, 6, 1, -10 }, // ')'
00113     { 61, 6, 6, 7, 1, -9 }, // '*'
00114     { 66, 8, 7, 11, 1, -7 }, // '+'
00115     { 73, 2, 3, 5, 1, -2 }, // ','
00116     { 74, 4, 1, 5, 1, -4 }, // '-'
00117     { 75, 2, 2, 5, 1, -2 }, // '.'
00118     { 76, 5, 10, 5, 0, -9 }, // '/'
00119     { 83, 7, 9, 9, 1, -9 }, // '0'
00120     { 91, 6, 9, 9, 1, -9 }, // '1'
00121     { 98, 7, 9, 9, 1, -9 }, // '2'
00122     { 106, 7, 9, 9, 1, -9 }, // '3'
00123     { 114, 7, 9, 9, 1, -9 }, // '4'
00124     { 122, 7, 9, 9, 1, -9 }, // '5'
00125     { 130, 7, 9, 9, 1, -9 }, // '6'
00126     { 138, 7, 9, 9, 1, -9 }, // '7'
00127     { 146, 7, 9, 9, 1, -9 }, // '8'
00128     { 154, 7, 9, 9, 1, -9 }, // '9'
00129     { 162, 2, 6, 5, 1, -6 }, // ':'
00130     { 164, 2, 7, 5, 1, -6 }, // ';'
00131     { 166, 9, 6, 11, 1, -7 }, // '<'
00132     { 173, 9, 3, 11, 1, -5 }, // '='
00133     { 177, 9, 6, 11, 1, -7 }, // '>'
00134     { 184, 6, 9, 7, 0, -9 }, // '?'
00135     { 191, 12, 11, 14, 1, -9 }, // '@'
00136     { 208, 9, 9, 9, 0, -9 }, // 'A'
00137     { 219, 7, 9, 9, 1, -9 }, // 'B'
00138     { 227, 7, 9, 9, 1, -9 }, // 'C'
00139     { 235, 8, 9, 10, 1, -9 }, // 'D'
00140     { 244, 7, 9, 9, 1, -9 }, // 'E'
00141     { 252, 6, 9, 8, 1, -9 }, // 'F'
00142     { 259, 8, 9, 10, 1, -9 }, // 'G'
00143     { 268, 8, 9, 10, 1, -9 }, // 'H'
00144     { 277, 2, 9, 4, 1, -9 }, // 'I'
00145     { 280, 4, 11, 4, -1, -9 }, // 'J'
00146     { 286, 8, 9, 8, 1, -9 }, // 'K'
00147     { 295, 6, 9, 7, 1, -9 }, // 'L'
00148     { 302, 9, 9, 11, 1, -9 }, // 'M'
00149     { 313, 8, 9, 10, 1, -9 }, // 'N'
00150     { 322, 8, 9, 10, 1, -9 }, // 'O'
00151     { 331, 7, 9, 9, 1, -9 }, // 'P'
00152     { 339, 8, 11, 10, 1, -9 }, // 'Q'
00153     { 350, 8, 9, 9, 1, -9 }, // 'R'
00154     { 359, 7, 9, 9, 1, -9 }, // 'S'
00155     { 367, 8, 9, 8, 0, -9 }, // 'T'
00156     { 376, 8, 9, 10, 1, -9 }, // 'U'
00157     { 385, 11, 9, 9, -1, -9 }, // 'V'
00158     { 398, 12, 9, 12, 0, -9 }, // 'W'
00159     { 412, 8, 9, 8, 0, -9 }, // 'X'
00160     { 421, 8, 9, 8, 0, -9 }, // 'Y'
00161     { 430, 8, 9, 10, 1, -9 }, // 'Z'
00162     { 439, 3, 11, 6, 2, -9 }, // '['
00163     { 444, 5, 10, 5, 0, -9 }, // '\'
00164     { 451, 3, 11, 6, 1, -9 }, // ']'
00165     { 456, 9, 3, 11, 1, -9 }, // '^'
00166     { 460, 7, 1, 7, 0, 2 }, // '_'
00167     { 461, 4, 2, 7, 1, -10 }, // '`'
00168     { 462, 7, 7, 9, 1, -7 }, // 'a'
00169     { 469, 7, 10, 9, 1, -10 }, // 'b'
00170     { 478, 6, 7, 8, 1, -7 }, // 'c'
00171     { 484, 7, 10, 9, 1, -10 }, // 'd'
00172     { 493, 7, 7, 9, 1, -7 }, // 'e'
00173     { 500, 5, 10, 5, 0, -10 }, // 'f'
00174     { 507, 7, 10, 9, 1, -7 }, // 'g'
00175     { 516, 7, 10, 9, 1, -10 }, // 'h'
00176     { 525, 2, 9, 4, 1, -9 }, // 'i'
00177     { 528, 3, 12, 4, 0, -9 }, // 'j'
00178     { 533, 7, 10, 8, 1, -10 }, // 'k'

```

```

00179     { 542, 2, 10, 4, 1, -10 }, // 'l'
00180     { 545, 10, 7, 12, 1, -7 }, // 'm'
00181     { 554, 7, 7, 9, 1, -7 }, // 'n'
00182     { 561, 7, 7, 9, 1, -7 }, // 'o'
00183     { 568, 7, 10, 9, 1, -7 }, // 'p'
00184     { 577, 7, 10, 9, 1, -7 }, // 'q'
00185     { 586, 5, 7, 6, 1, -7 }, // 'r'
00186     { 591, 6, 7, 8, 1, -7 }, // 's'
00187     { 597, 5, 9, 6, 0, -9 }, // 't'
00188     { 603, 7, 7, 9, 1, -7 }, // 'u'
00189     { 610, 7, 7, 7, 0, -7 }, // 'v'
00190     { 617, 10, 7, 10, 0, -7 }, // 'w'
00191     { 626, 7, 7, 7, 0, -7 }, // 'x'
00192     { 633, 7, 10, 7, 0, -7 }, // 'y'
00193     { 642, 6, 7, 6, 0, -7 }, // 'z'
00194     { 648, 6, 11, 9, 2, -9 }, // '{'
00195     { 657, 2, 12, 5, 2, -9 }, // '|'
00196     { 660, 6, 11, 9, 1, -9 }, // '}'
00197 };
00198 const GFXfont SansSerif_plain_12 PROGMEM = {
00199 (uint8_t *)SansSerif_plain_12Bitmaps, (GFXglyph *)SansSerif_plain_12Glyphs, 0x20, 0x7E, 15};

```

## 2.18 SansSerif9.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t SansSerif_plain_9Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA, 0x88, // '!'
00008     0xAA, // '"'
00009     0x28, 0x53, 0xF1, 0x4F, 0xCA, 0x14, 0x00, // '#'
00010     0x21, 0xEA, 0x38, 0x38, 0xAF, 0x08, // '$'
00011     0xE4, 0x52, 0x2A, 0x1F, 0xE1, 0x51, 0x28, 0x9C, // '%'
00012     0x30, 0x91, 0x03, 0x29, 0x53, 0x1B, 0x00, // '&'
00013     0xA0, // "'"
00014     0x52, 0x49, 0x22, // '('
00015     0x91, 0x24, 0xA4, // ')'
00016     0xA9, 0xC7, 0x2A, // '*'
00017     0x20, 0x8F, 0x88, 0x20, // '+'
00018     0xA0, // ','
00019     0xC0, // '-'
00020     0x80, // '.'
00021     0x22, 0x44, 0x48, 0x80, // '/'
00022     0x64, 0xA5, 0x29, 0x49, 0x80, // '0'
00023     0xC4, 0x44, 0x44, 0xE0, // '1'
00024     0x64, 0x84, 0x44, 0x43, 0xC0, // '2'
00025     0x64, 0x84, 0xC1, 0x0B, 0x80, // '3'
00026     0x10, 0xC5, 0x14, 0x93, 0xE1, 0x00, // '4'
00027     0xF4, 0x21, 0xC1, 0x0B, 0x80, // '5'
00028     0x76, 0x21, 0xC9, 0x49, 0x80, // '6'
00029     0xF0, 0x88, 0x42, 0x21, 0x00, // '7'
00030     0x64, 0xA4, 0xC9, 0x49, 0x80, // '8'
00031     0x64, 0xA5, 0xE1, 0x1B, 0x80, // '9'
00032     0x80, 0x80, // ':'
00033     0x80, 0xA0, // ';'
00034     0x04, 0x73, 0x01, 0xC0, 0x40, // '<'
00035     0xFC, 0x03, 0xF0, // '='
00036     0x80, 0xE0, 0x33, 0x88, 0x00, // '>'
00037     0xF0, 0x88, 0x84, 0x01, 0x00, // '?'
00038     0x3C, 0x21, 0x27, 0x54, 0xAA, 0x54, 0xF1, 0x10, 0x70, // '@'
00039     0x30, 0x61, 0x22, 0x47, 0x90, 0xA1, 0x00, // 'A'
00040     0xF2, 0x28, 0xBC, 0x8A, 0x2F, 0x00, // 'B'
00041     0x73, 0x28, 0x20, 0x83, 0x07, 0x80, // 'C'
00042     0xF2, 0x68, 0xA2, 0x8A, 0x6F, 0x00, // 'D'
00043     0xF4, 0x21, 0xE8, 0x43, 0xC0, // 'E'
00044     0xF4, 0x21, 0xE8, 0x42, 0x00, // 'F'
00045     0x73, 0x28, 0x26, 0x8B, 0x27, 0x00, // 'G'
00046     0x8A, 0x28, 0xBE, 0x8A, 0x28, 0x80, // 'H'
00047     0xAA, 0xA8, // 'I'
00048     0x49, 0x24, 0x92, 0x80, // 'J'
00049     0x8A, 0x4A, 0x30, 0xA2, 0x48, 0x80, // 'K'
00050     0x84, 0x21, 0x08, 0x43, 0xC0, // 'L'
00051     0x85, 0x9B, 0x35, 0xAB, 0x50, 0xA1, 0x00, // 'M'
00052     0x8B, 0x2C, 0xAA, 0x9A, 0x68, 0x80, // 'N'
00053     0x73, 0x68, 0xA2, 0x8B, 0x67, 0x00, // 'O'
00054     0xE4, 0xA5, 0xC8, 0x42, 0x00, // 'P'
00055     0x73, 0x68, 0xA2, 0x8B, 0x67, 0x04, // 'Q'
00056     0xF2, 0x49, 0x38, 0xA2, 0x48, 0x80, // 'R'
00057     0x72, 0x28, 0x1C, 0x0A, 0x27, 0x00, // 'S'
00058     0xF8, 0x82, 0x08, 0x20, 0x82, 0x00, // 'T'
00059     0x8A, 0x28, 0xA2, 0x8A, 0x27, 0x00, // 'U'

```

```

00060      0x85,0x09,0x22,0x44,0x86,0x0C,0x00, // 'V'
00061      0x92,0x92,0x54,0x54,0x6C,0x28,0x28, // 'W'
00062      0xCC,0x90,0xC1,0x83,0x09,0x23,0x00, // 'X'
00063      0x89,0x45,0x08,0x20,0x82,0x00, // 'Y'
00064      0xF8,0x21,0x08,0x42,0x0F,0x80, // 'Z'
00065      0xD2,0x49,0x26, // '['
00066      0x88,0x44,0x42,0x20, // '\
00067      0xC9,0x24,0x96, // ']'
00068      0x30,0x90, // '^'
00069      0xF8, // '._'
00070      0x88, // '``'
00071      0x70,0xBD,0x2F,0x00, // 'a'
00072      0x84,0x21,0xC9,0x4A,0x5C, // 'b'
00073      0x74,0x21,0x07,0x00, // 'c'
00074      0x10,0x84,0xE9,0x4A,0x4E, // 'd'
00075      0x64,0xBD,0x07,0x00, // 'e'
00076      0x72,0x11,0xC4,0x21,0x08, // 'f'
00077      0x74,0xA5,0x27,0x09,0x80, // 'g'
00078      0x84,0x21,0xE9,0x4A,0x52, // 'h'
00079      0x8A,0xA8, // 'i'
00080      0x41,0x24,0x92,0xC0, // 'j'
00081      0x84,0x21,0x2A,0x62,0x92, // 'k'
00082      0xAA,0xAA, // 'l'
00083      0xFE,0x92,0x92,0x92,0x92, // 'm'
00084      0xF4,0xA5,0x29,0x00, // 'n'
00085      0x64,0xA5,0x26,0x00, // 'o'
00086      0xE4,0xA5,0x2E,0x42,0x00, // 'p'
00087      0x74,0xA5,0x27,0x08,0x40, // 'q'
00088      0xE8,0x88,0x80, // 'r'
00089      0xE8,0x62,0xE0, // 's'
00090      0x47,0x90,0x84,0x38, // 't'
00091      0x94,0xA5,0x2F,0x00, // 'u'
00092      0x8A,0x25,0x14,0x20, // 'v'
00093      0x92,0xAA,0xAA,0x44,0x44, // 'w'
00094      0x89,0x42,0x14,0x88, // 'x'
00095      0x89,0x11,0x42,0x82,0x04,0x30,0x00, // 'y'
00096      0xF0,0x88,0x8F,0x00, // 'z'
00097      0x64,0x48,0x44,0x46, // '{'
00098      0xAA,0xAA,0x80, // '|'
00099      0xC4,0x42,0x44,0x4C // '}'
00100  };
00101  const GFXglyph SansSerif_plain_9Glyphs[] PROGMEM = {
00102  // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103      { 0, 2, 1, 4, 0, -1 }, // ' '
00104      { 1, 2, 7, 4, 1, -7 }, // '!'
00105      { 3, 4, 2, 5, 1, -7 }, // '"'
00106      { 4, 7, 7, 9, 1, -7 }, // '#'
00107      { 11, 6, 8, 7, 0, -7 }, // '$'
00108      { 17, 9, 7, 10, 0, -7 }, // '%'
00109      { 25, 7, 7, 9, 1, -7 }, // '&'
00110      { 32, 2, 2, 3, 1, -7 }, // "'"
00111      { 33, 3, 8, 5, 1, -8 }, // '('
00112      { 36, 3, 8, 5, 1, -8 }, // ')'
00113      { 39, 6, 4, 6, 0, -7 }, // '*'
00114      { 42, 6, 5, 9, 1, -5 }, // '+'
00115      { 46, 2, 2, 4, 1, -1 }, // ','
00116      { 47, 3, 1, 4, 1, -3 }, // '-'
00117      { 48, 2, 1, 4, 1, -1 }, // '.'
00118      { 49, 4, 7, 4, 0, -7 }, // '/'
00119      { 53, 5, 7, 7, 1, -7 }, // '0'
00120      { 58, 4, 7, 7, 2, -7 }, // '1'
00121      { 62, 5, 7, 7, 1, -7 }, // '2'
00122      { 67, 5, 7, 7, 1, -7 }, // '3'
00123      { 72, 6, 7, 7, 1, -7 }, // '4'
00124      { 78, 5, 7, 7, 1, -7 }, // '5'
00125      { 83, 5, 7, 7, 1, -7 }, // '6'
00126      { 88, 5, 7, 7, 1, -7 }, // '7'
00127      { 93, 5, 7, 7, 1, -7 }, // '8'
00128      { 98, 5, 7, 7, 1, -7 }, // '9'
00129      { 103, 2, 5, 4, 1, -5 }, // ':'
00130      { 105, 2, 6, 4, 1, -5 }, // ';'
00131      { 107, 7, 5, 9, 1, -5 }, // '<'
00132      { 112, 7, 3, 9, 1, -4 }, // '='
00133      { 115, 7, 5, 9, 1, -5 }, // '>'
00134      { 120, 5, 7, 6, 1, -7 }, // '?'
00135      { 125, 9, 8, 11, 1, -7 }, // '@'
00136      { 134, 7, 7, 7, 0, -7 }, // 'A'
00137      { 141, 6, 7, 8, 1, -7 }, // 'B'
00138      { 147, 6, 7, 8, 1, -7 }, // 'C'
00139      { 153, 6, 7, 8, 1, -7 }, // 'D'
00140      { 159, 5, 7, 7, 1, -7 }, // 'E'
00141      { 164, 5, 7, 7, 1, -7 }, // 'F'
00142      { 169, 6, 7, 8, 1, -7 }, // 'G'
00143      { 175, 6, 7, 8, 1, -7 }, // 'H'
00144      { 181, 2, 7, 4, 1, -7 }, // 'I'
00145      { 183, 3, 9, 4, 0, -7 }, // 'J'
00146      { 187, 6, 7, 7, 1, -7 }, // 'K'

```

```

00147     { 193, 5, 7, 6, 1, -7 }, // 'L'
00148     { 198, 7, 7, 9, 1, -7 }, // 'M'
00149     { 205, 6, 7, 8, 1, -7 }, // 'N'
00150     { 211, 6, 7, 8, 1, -7 }, // 'O'
00151     { 217, 5, 7, 7, 1, -7 }, // 'P'
00152     { 222, 6, 8, 8, 1, -7 }, // 'Q'
00153     { 228, 6, 7, 7, 1, -7 }, // 'R'
00154     { 234, 6, 7, 8, 1, -7 }, // 'S'
00155     { 240, 6, 7, 6, 0, -7 }, // 'T'
00156     { 246, 6, 7, 8, 1, -7 }, // 'U'
00157     { 252, 7, 7, 7, 0, -7 }, // 'V'
00158     { 259, 8, 7, 8, 0, -7 }, // 'W'
00159     { 266, 7, 7, 7, 0, -7 }, // 'X'
00160     { 273, 6, 7, 6, 0, -7 }, // 'Y'
00161     { 279, 6, 7, 6, 0, -7 }, // 'Z'
00162     { 285, 3, 8, 5, 1, -7 }, // '['
00163     { 288, 4, 7, 4, 0, -7 }, // '\'
00164     { 292, 3, 8, 5, 1, -7 }, // ']'
00165     { 295, 7, 2, 9, 1, -7 }, // '^'
00166     { 297, 6, 1, 6, 0, 1 }, // '_'
00167     { 298, 3, 2, 6, 1, -8 }, // '`'
00168     { 299, 5, 5, 7, 1, -5 }, // 'a'
00169     { 303, 5, 8, 7, 1, -8 }, // 'b'
00170     { 308, 5, 5, 7, 1, -5 }, // 'c'
00171     { 312, 5, 8, 7, 1, -8 }, // 'd'
00172     { 317, 5, 5, 7, 1, -5 }, // 'e'
00173     { 321, 5, 8, 4, 0, -8 }, // 'f'
00174     { 326, 5, 7, 7, 1, -5 }, // 'g'
00175     { 331, 5, 8, 7, 1, -8 }, // 'h'
00176     { 336, 2, 7, 4, 1, -7 }, // 'i'
00177     { 338, 3, 9, 4, 0, -7 }, // 'j'
00178     { 342, 5, 8, 6, 1, -8 }, // 'k'
00179     { 347, 2, 8, 4, 1, -8 }, // 'l'
00180     { 349, 8, 5, 10, 1, -5 }, // 'm'
00181     { 354, 5, 5, 7, 1, -5 }, // 'n'
00182     { 358, 5, 5, 7, 1, -5 }, // 'o'
00183     { 362, 5, 7, 7, 1, -5 }, // 'p'
00184     { 367, 5, 7, 7, 1, -5 }, // 'q'
00185     { 372, 4, 5, 5, 1, -5 }, // 'r'
00186     { 375, 4, 5, 6, 1, -5 }, // 's'
00187     { 378, 5, 6, 5, 0, -6 }, // 't'
00188     { 382, 5, 5, 7, 1, -5 }, // 'u'
00189     { 386, 6, 5, 6, 0, -5 }, // 'v'
00190     { 390, 8, 5, 8, 0, -5 }, // 'w'
00191     { 395, 6, 5, 6, 0, -5 }, // 'x'
00192     { 399, 7, 7, 6, 0, -5 }, // 'y'
00193     { 406, 5, 5, 7, 1, -5 }, // 'z'
00194     { 410, 4, 8, 6, 1, -7 }, // '{'
00195     { 414, 2, 9, 4, 1, -7 }, // '|'
00196     { 417, 4, 8, 6, 1, -7 }, // '}'
00197 };
00198 const GFXfont SansSerif_plain_9 PROGMEM = {
00199 (uint8_t *)SansSerif_plain_9Bitmaps, (GFXglyph *)SansSerif_plain_9Glyphs, 0x20, 0x7E, 12};

```

## 2.19 serif12.h

```

00001 // Created by https://oledisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Serif_plain_12Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xDB, 0x6D, 0x86, 0xC0, // '!'
00008     0xAA, 0xA0, // '"'
00009     0x12, 0x0A, 0x1F, 0xC2, 0x82, 0x41, 0x23, 0xF8, 0x50, 0x48, 0x00, // '#'
00010     0x20, 0x41, 0xE5, 0x6A, 0x4C, 0x0E, 0x52, 0xA4, 0xF0, 0x81, 0x00, // '$'
00011     0x61, 0x12, 0x42, 0x48, 0x4A, 0x06, 0xD8, 0x14, 0x84, 0x90, 0x92, 0x21, 0x80, // '%'
00012     0x38, 0x11, 0x04, 0x01, 0x80, 0x93, 0xA2, 0x48, 0x63, 0x08, 0x7D, 0x80, // '&'
00013     0xA8, // '''
00014     0x24, 0x88, 0x88, 0x88, 0x84, 0x20, // '('
00015     0x84, 0x42, 0x22, 0x22, 0x44, 0x80, // ')'
00016     0x22, 0xA7, 0x3E, 0x20, // '*'
00017     0x10, 0x10, 0x10, 0xFE, 0x10, 0x10, 0x10, // '+'
00018     0x4A, 0x00, // ','
00019     0xE0, // '-'
00020     0xD8, // '.'
00021     0x10, 0x88, 0x42, 0x21, 0x08, 0x84, 0x00, // '/'
00022     0x78, 0x92, 0x14, 0x28, 0x50, 0xA1, 0x24, 0x78, // '0'
00023     0x62, 0x82, 0x08, 0x20, 0x82, 0x08, 0xF8, // '1'
00024     0x79, 0x08, 0x10, 0x20, 0x82, 0x08, 0x22, 0xFC, // '2'
00025     0x79, 0x08, 0x10, 0x23, 0x81, 0x81, 0x42, 0x78, // '3'
00026     0x08, 0x18, 0x28, 0x48, 0x48, 0xFE, 0x08, 0x08, 0x1C, // '4'
00027     0xF9, 0x02, 0x07, 0xC8, 0xC0, 0x81, 0x46, 0x78, // '5'

```

```

00028      0x38, 0x8A, 0x05, 0xCC, 0xD0, 0xA1, 0x26, 0x78, // '6'
00029      0xFD, 0x08, 0x20, 0x41, 0x02, 0x04, 0x10, 0x20, // '7'
00030      0x79, 0x0A, 0x14, 0x27, 0x99, 0xA1, 0x42, 0x78, // '8'
00031      0x79, 0x92, 0x14, 0x2C, 0xCE, 0x81, 0x44, 0x70, // '9'
00032      0xD8, 0x6C, // ':'
00033      0xD8, 0x04, 0xA0, // ';'
00034      0x03, 0x0E, 0x38, 0x1C, 0x01, 0xC0, 0x18, // '<'
00035      0xFF, 0x00, 0x3F, 0xC0, // '='
00036      0xC0, 0x1C, 0x01, 0xC0, 0xE3, 0x86, 0x00, // '>'
00037      0x73, 0x28, 0x86, 0x30, 0x80, 0x0C, 0x30, // '?'
00038      0x1E, 0x04, 0x31, 0x03, 0x47, 0x29, 0x25, 0x24, 0xA4, 0xA4, 0x78, 0x40, 0x0C, 0x20, 0x78, 0x00, // '@'
00039      0x08, 0x07, 0x01, 0x40, 0x50, 0x22, 0x08, 0x87, 0xF1, 0x04, 0xE3, 0x80, // 'A'
00040      0xFC, 0x42, 0x42, 0x42, 0x7C, 0x42, 0x42, 0x42, 0xFC, // 'B'
00041      0x3E, 0x20, 0xA0, 0x10, 0x08, 0x04, 0x02, 0x00, 0x86, 0x3E, 0x00, // 'C'
00042      0xFC, 0x23, 0x10, 0x48, 0x24, 0x12, 0x09, 0x04, 0x8C, 0xFC, 0x00, // 'D'
00043      0xFE, 0x42, 0x40, 0x44, 0x7C, 0x44, 0x40, 0x42, 0xFE, // 'E'
00044      0xFE, 0x42, 0x40, 0x44, 0x7C, 0x44, 0x40, 0x40, 0xE0, // 'F'
00045      0x3E, 0x20, 0xA0, 0x10, 0x08, 0x04, 0x3A, 0x04, 0x82, 0x3E, 0x00, // 'G'
00046      0xE3, 0x90, 0x44, 0x11, 0x04, 0x7F, 0x10, 0x44, 0x11, 0x04, 0xE3, 0x80, // 'H'
00047      0xE4, 0x44, 0x44, 0x44, 0xE0, // 'I'
00048      0x38, 0x41, 0x04, 0x10, 0x41, 0x04, 0x10, 0x49, 0x38, // 'J'
00049      0xEE, 0x22, 0x12, 0x0A, 0x06, 0x02, 0x81, 0x20, 0x88, 0xE3, 0x00, // 'K'
00050      0xE0, 0x40, 0x40, 0x40, 0x40, 0x40, 0x40, 0x42, 0xFE, // 'L'
00051      0xE0, 0xE6, 0x0C, 0x51, 0x45, 0x14, 0x4A, 0x44, 0xA4, 0x44, 0x44, 0x04, 0xE0, 0xE0, // 'M'
00052      0xC3, 0x98, 0x45, 0x11, 0x64, 0x49, 0x11, 0x44, 0x31, 0x0C, 0xE1, 0x00, // 'N'
00053      0x3E, 0x10, 0x48, 0x0A, 0x02, 0x80, 0xA0, 0x28, 0x09, 0x04, 0x3E, 0x00, // 'O'
00054      0xFC, 0x46, 0x42, 0x46, 0x7C, 0x40, 0x40, 0x40, 0x40, 0xE0, // 'P'
00055      0x3E, 0x10, 0x48, 0x0A, 0x02, 0x80, 0xA0, 0x28, 0x09, 0x04, 0x3E, 0x01, 0x00, 0x30, // 'Q'
00056      0xFC, 0x10, 0x84, 0x21, 0x08, 0x7C, 0x11, 0x04, 0x21, 0x08, 0xE1, 0x80, // 'R'
00057      0x79, 0x0A, 0x06, 0x03, 0x01, 0x81, 0x42, 0x78, // 'S'
00058      0xFE, 0x92, 0x10, 0x10, 0x10, 0x10, 0x10, 0x38, // 'T'
00059      0xE3, 0x90, 0x44, 0x11, 0x04, 0x41, 0x10, 0x44, 0x11, 0x8C, 0x3E, 0x00, // 'U'
00060      0xE3, 0x90, 0x44, 0x30, 0x88, 0x22, 0x05, 0x01, 0x40, 0x70, 0x08, 0x00, // 'V'
00061      0xE6, 0x72, 0x31, 0x11, 0x88, 0x52, 0x82, 0x94, 0x14, 0xA0, 0xA5, 0x02, 0x10, 0x10, 0x80, // 'W'
00062      0xE7, 0x21, 0x09, 0x03, 0x01, 0x80, 0xC0, 0x90, 0x84, 0xE7, 0x00, // 'X'
00063      0xC3, 0x90, 0x42, 0x20, 0x50, 0x08, 0x02, 0x00, 0x80, 0x20, 0x1C, 0x00, // 'Y'
00064      0xFE, 0x84, 0x0C, 0x08, 0x10, 0x20, 0x60, 0x42, 0xFE, // 'Z'
00065      0xE8, 0x88, 0x88, 0x88, 0xE0, // '['
00066      0x84, 0x10, 0x84, 0x10, 0x84, 0x10, 0x80, // '\'
00067      0xE2, 0x22, 0x22, 0x22, 0xE0, // ']'
00068      0x18, 0x12, 0x10, 0x80, // '^'
00069      0xFC, // '_'
00070      0x42, // `
00071      0x71, 0x10, 0x23, 0xC8, 0x91, 0x1F, 0x00, // 'a'
00072      0xC0, 0x40, 0x40, 0x5C, 0x66, 0x42, 0x42, 0x66, 0xDC, // 'b'
00073      0x79, 0x8A, 0x04, 0x08, 0x18, 0x9E, 0x00, // 'c'
00074      0x0C, 0x04, 0x04, 0x74, 0xCC, 0x84, 0x84, 0x84, 0xCC, 0x76, // 'd'
00075      0x79, 0x92, 0x17, 0xE8, 0x18, 0x9E, 0x00, // 'e'
00076      0x39, 0x04, 0x3C, 0x41, 0x04, 0x10, 0x43, 0x80, // 'f'
00077      0x76, 0xCC, 0x84, 0x84, 0x84, 0xCC, 0x74, 0x04, 0x8C, 0x78, // 'g'
00078      0xC0, 0x40, 0x40, 0x58, 0x64, 0x44, 0x44, 0x44, 0x44, 0xE0, // 'h'
00079      0x40, 0xC4, 0x44, 0x44, 0xE0, // 'i'
00080      0x10, 0x0C, 0x21, 0x08, 0x42, 0x10, 0x85, 0xE0, // 'j'
00081      0xC0, 0x40, 0x40, 0x4C, 0x48, 0x50, 0x70, 0x48, 0x44, 0xEE, // 'k'
00082      0xC4, 0x44, 0x44, 0x44, 0x4E, // 'l'
00083      0xD9, 0x86, 0x64, 0x44, 0x44, 0x44, 0x44, 0x44, 0xEE, 0xE0, // 'm'
00084      0xD8, 0x64, 0x44, 0x44, 0x44, 0x44, 0x44, 0xEE, // 'n'
00085      0x79, 0x9A, 0x14, 0x28, 0x59, 0x9E, 0x00, // 'o'
00086      0xDC, 0x66, 0x42, 0x42, 0x66, 0x5C, 0x40, 0x40, 0xE0, // 'p'
00087      0x76, 0xCC, 0x84, 0x84, 0x84, 0xCC, 0x74, 0x04, 0x04, 0x0E, // 'q'
00088      0xD9, 0xA4, 0x10, 0x41, 0x0E, 0x00, // 'r'
00089      0x72, 0x28, 0x1C, 0x0A, 0x27, 0x00, // 's'
00090      0x42, 0x3C, 0x84, 0x21, 0x0A, 0x70, // 't'
00091      0xCC, 0x44, 0x44, 0x44, 0x44, 0x3E, // 'u'
00092      0xEE, 0x44, 0x44, 0x28, 0x28, 0x10, 0x10, // 'v'
00093      0xE4, 0xE4, 0x44, 0x44, 0x42, 0xA8, 0x2A, 0x81, 0x10, 0x11, 0x00, // 'w'
00094      0xEC, 0x90, 0xC1, 0x83, 0x09, 0x37, 0x00, // 'x'
00095      0xEE, 0x44, 0x44, 0x68, 0x28, 0x28, 0x10, 0x10, 0x30, 0xE0, // 'y'
00096      0xFA, 0x41, 0x08, 0x41, 0x2F, 0x80, // 'z'
00097      0x38, 0x82, 0x08, 0x23, 0x02, 0x08, 0x20, 0x83, 0x80, // '{'
00098      0xAA, 0xAA, 0xAA, // '|'
00099      0xE0, 0x82, 0x08, 0x20, 0x62, 0x08, 0x20, 0x8E, 0x00 // '}'
00100 };
00101 const GFXglyph Serif_plain_12Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103 { 0, 2, 1, 5, 0, -1 }, // ' '
00104 { 1, 3, 9, 6, 2, -9 }, // '!'
00105 { 5, 4, 3, 7, 1, -9 }, // '"'
00106 { 7, 9, 9, 11, 1, -9 }, // '#'
00107 { 18, 7, 12, 9, 1, -10 }, // '$'
00108 { 29, 11, 9, 12, 1, -9 }, // '%'
00109 { 42, 10, 9, 12, 1, -9 }, // '&'
00110 { 54, 2, 3, 4, 1, -9 }, // "'"
00111 { 55, 4, 11, 6, 1, -10 }, // '('
00112 { 61, 4, 11, 6, 1, -10 }, // ')'
00113 { 67, 6, 5, 7, 1, -9 }, // '*'
00114 { 71, 8, 7, 11, 2, -7 }, // '+'

```

```

00115 { 78, 3, 3, 5, 1, -1 }, // ','
00116 { 80, 4, 1, 5, 1, -4 }, // '-'
00117 { 81, 3, 2, 5, 1, -2 }, // '.'
00118 { 82, 5, 10, 5, 0, -9 }, // '/'
00119 { 89, 7, 9, 9, 1, -9 }, // '0'
00120 { 97, 6, 9, 9, 1, -9 }, // '1'
00121 { 104, 7, 9, 9, 1, -9 }, // '2'
00122 { 112, 7, 9, 9, 1, -9 }, // '3'
00123 { 120, 8, 9, 9, 1, -9 }, // '4'
00124 { 129, 7, 9, 9, 1, -9 }, // '5'
00125 { 137, 7, 9, 9, 1, -9 }, // '6'
00126 { 145, 7, 9, 9, 1, -9 }, // '7'
00127 { 153, 7, 9, 9, 1, -9 }, // '8'
00128 { 161, 7, 9, 9, 1, -9 }, // '9'
00129 { 169, 3, 5, 5, 1, -5 }, // ':'
00130 { 171, 3, 7, 5, 1, -5 }, // ';'
00131 { 174, 9, 6, 11, 1, -7 }, // '<'
00132 { 181, 9, 3, 11, 1, -5 }, // '='
00133 { 185, 9, 6, 11, 1, -7 }, // '>'
00134 { 192, 6, 9, 7, 1, -9 }, // '?'
00135 { 199, 11, 11, 13, 1, -8 }, // '@'
00136 { 215, 10, 9, 10, 0, -9 }, // 'A'
00137 { 227, 8, 9, 10, 1, -9 }, // 'B'
00138 { 236, 9, 9, 10, 1, -9 }, // 'C'
00139 { 247, 9, 9, 10, 1, -9 }, // 'D'
00140 { 258, 8, 9, 9, 1, -9 }, // 'E'
00141 { 267, 8, 9, 9, 1, -9 }, // 'F'
00142 { 276, 9, 9, 10, 1, -9 }, // 'G'
00143 { 287, 10, 9, 11, 1, -9 }, // 'H'
00144 { 299, 4, 9, 5, 1, -9 }, // 'I'
00145 { 304, 6, 12, 5, -1, -9 }, // 'J'
00146 { 313, 9, 9, 10, 1, -9 }, // 'K'
00147 { 324, 8, 9, 9, 1, -9 }, // 'L'
00148 { 333, 12, 9, 13, 1, -9 }, // 'M'
00149 { 347, 10, 9, 11, 1, -9 }, // 'N'
00150 { 359, 10, 9, 11, 1, -9 }, // 'O'
00151 { 371, 8, 9, 9, 1, -9 }, // 'P'
00152 { 380, 10, 11, 11, 1, -9 }, // 'Q'
00153 { 394, 10, 9, 10, 1, -9 }, // 'R'
00154 { 406, 7, 9, 8, 1, -9 }, // 'S'
00155 { 414, 8, 9, 9, 1, -9 }, // 'T'
00156 { 423, 10, 9, 11, 1, -9 }, // 'U'
00157 { 435, 10, 9, 10, 0, -9 }, // 'V'
00158 { 447, 13, 9, 13, 0, -9 }, // 'W'
00159 { 462, 9, 9, 10, 1, -9 }, // 'X'
00160 { 473, 10, 9, 9, 0, -9 }, // 'Y'
00161 { 485, 8, 9, 9, 1, -9 }, // 'Z'
00162 { 494, 4, 11, 6, 1, -10 }, // '['
00163 { 500, 5, 10, 5, 0, -9 }, // '\'
00164 { 507, 4, 11, 6, 1, -10 }, // ']'
00165 { 513, 9, 3, 11, 1, -9 }, // '^'
00166 { 517, 7, 1, 7, 0, 2 }, // '_'
00167 { 518, 4, 2, 7, 1, -10 }, // '`'
00168 { 519, 7, 7, 8, 1, -7 }, // 'a'
00169 { 526, 8, 10, 9, 1, -10 }, // 'b'
00170 { 536, 7, 7, 8, 1, -7 }, // 'c'
00171 { 543, 8, 10, 9, 1, -10 }, // 'd'
00172 { 553, 7, 7, 8, 1, -7 }, // 'e'
00173 { 560, 6, 10, 6, 1, -10 }, // 'f'
00174 { 568, 8, 10, 9, 1, -7 }, // 'g'
00175 { 578, 8, 10, 9, 1, -10 }, // 'h'
00176 { 588, 4, 9, 5, 1, -9 }, // 'i'
00177 { 593, 5, 12, 5, -1, -9 }, // 'j'
00178 { 601, 8, 10, 9, 1, -10 }, // 'k'
00179 { 611, 4, 10, 5, 1, -10 }, // 'l'
00180 { 616, 12, 7, 13, 1, -7 }, // 'm'
00181 { 627, 8, 7, 9, 1, -7 }, // 'n'
00182 { 634, 7, 7, 8, 1, -7 }, // 'o'
00183 { 641, 8, 10, 9, 1, -7 }, // 'p'
00184 { 651, 8, 10, 9, 1, -7 }, // 'q'
00185 { 661, 6, 7, 7, 1, -7 }, // 'r'
00186 { 667, 6, 7, 7, 1, -7 }, // 's'
00187 { 673, 5, 9, 6, 1, -9 }, // 't'
00188 { 679, 8, 7, 9, 1, -7 }, // 'u'
00189 { 686, 8, 7, 7, 0, -7 }, // 'v'
00190 { 693, 12, 7, 11, 0, -7 }, // 'w'
00191 { 704, 7, 7, 8, 1, -7 }, // 'x'
00192 { 711, 8, 10, 8, 0, -7 }, // 'y'
00193 { 721, 6, 7, 7, 1, -7 }, // 'z'
00194 { 727, 6, 11, 9, 1, -10 }, // '{'
00195 { 736, 2, 12, 5, 2, -9 }, // '|'
00196 { 739, 6, 11, 9, 1, -10 }, // '}'
00197 };
00198 const GFXfont Serif_plain_12 PROGMEM = {
00199 (uint8_t *)Serif_plain_12Bitmaps, (GFXglyph *)Serif_plain_12Glyphs, 0x20, 0x7E, 15};

```



## 2.20 serif9.h

```

00001 // Created by https://oleddisplay.squix.ch/ Consider a donation
00002 // In case of problems make sure that you are using the font file with the correct version!
00003 const uint8_t Serif_plain_9Bitmaps[] PROGMEM = {
00004
00005     // Bitmap Data:
00006     0x00, // ' '
00007     0xAA,0x88, // '!'
00008     0xAA, // '"'
00009     0x28,0x51,0xF1,0x8F,0xCA,0x00, // '#'
00010     0x47,0xB1,0xC7,0x2B,0x88, // '$'
00011     0xE4,0x52,0x2A,0x1F,0xE1,0x51,0x28,0x9C, // '%'
00012     0x70,0x50,0x40,0xAE,0xB4,0x98,0x76, // '&'
00013     0xA0, // "'"
00014     0x52,0x49,0x22, // '('
00015     0x89,0x24,0x94, // ')'
00016     0x23,0xEF,0x88, // '*'
00017     0x20,0x8F,0x88,0x20, // '+'
00018     0x50, // ','
00019     0xC0, // '-'
00020     0x80, // '.'
00021     0x22,0x44,0x48,0x80, // '/'
00022     0x72,0x28,0xA2,0x8A,0x27,0x00, // '0'
00023     0x4C,0x44,0x44,0xE0, // '1'
00024     0x64,0x84,0x44,0x2B,0xC0, // '2'
00025     0x64,0x84,0x41,0x49,0x80, // '3'
00026     0x10,0xC5,0x14,0xF8,0x43,0x80, // '4'
00027     0xF4,0x38,0x21,0x49,0x80, // '5'
00028     0x73,0x28,0x3C,0x8A,0x27,0x00, // '6'
00029     0xF4,0x88,0x42,0x21,0x00, // '7'
00030     0x72,0x28,0x9C,0xDA,0x27,0x00, // '8'
00031     0x72,0x28,0x9E,0x0A,0x67,0x00, // '9'
00032     0x82, // ':'
00033     0x40,0x28, // ';'
00034     0x04,0x73,0x03,0x80,0xC0, // '<'
00035     0xFC,0x03,0xF0, // '='
00036     0x80,0xE0,0x31,0xCC,0x00, // '>'
00037     0xE0,0x84,0xC4,0x01,0x00, // '?'
00038     0x3C,0x21,0x27,0x54,0xAA,0x54,0xF1,0x00,0x78, // '@'
00039     0x10,0x30,0x28,0x48,0x7C,0x44,0xC6, // 'A'
00040     0xF8,0x89,0x13,0xC4,0x48,0xBE,0x00, // 'B'
00041     0x38,0x8A,0x04,0x08,0x08,0x9E,0x00, // 'C'
00042     0xF8,0x99,0x12,0x24,0x49,0xBE,0x00, // 'D'
00043     0xFC,0x89,0x03,0xC4,0x08,0xBF,0x00, // 'E'
00044     0xFC,0x89,0x03,0xC4,0x08,0x38,0x00, // 'F'
00045     0x38,0x8A,0x04,0x08,0xD8,0x9E,0x00, // 'G'
00046     0xEE,0x44,0x44,0x7C,0x44,0x44,0xEE, // 'H'
00047     0xE4,0x44,0x44,0xE0, // 'I'
00048     0x71,0x08,0x42,0x10,0x94,0xC0, // 'J'
00049     0xEC,0x48,0x50,0x60,0x50,0x48,0xE6, // 'K'
00050     0xE1,0x04,0x10,0x41,0x2F,0x80, // 'L'
00051     0xE3,0x98,0xC5,0x51,0x54,0x49,0x12,0x4E,0x38, // 'M'
00052     0xCE,0x64,0x64,0x54,0x4C,0x4C,0xE4, // 'N'
00053     0x79,0x9A,0x14,0x28,0x59,0x9E,0x00, // 'O'
00054     0xF1,0x24,0x9C,0x41,0x0E,0x00, // 'P'
00055     0x79,0x9A,0x14,0x28,0x59,0x1E,0x04, // 'Q'
00056     0xF8,0x91,0x23,0x85,0x09,0x39,0x00, // 'R'
00057     0x72,0x28,0x1C,0x0A,0x27,0x00, // 'S'
00058     0xFA,0xA2,0x08,0x20,0x87,0x00, // 'T'
00059     0xEE,0x44,0x44,0x44,0x44,0x38, // 'U'
00060     0xCE,0x44,0x44,0x28,0x28,0x10,0x10, // 'V'
00061     0xEB,0x92,0x45,0x51,0x54,0x36,0x08,0x82,0x20, // 'W'
00062     0xDC,0x90,0xC1,0x83,0x09,0x3B,0x00, // 'X'
00063     0xEE,0x44,0x28,0x10,0x10,0x10,0x38, // 'Y'
00064     0xFA,0x21,0x08,0x42,0x2F,0x80, // 'Z'
00065     0xD2,0x49,0x26, // '['
00066     0x88,0x44,0x42,0x20, // '\'
00067     0xC9,0x24,0x96, // ']'
00068     0x30,0x90, // '^'
00069     0xF8, // '_'
00070     0x88, // '`'
00071     0x60,0x4F,0x24,0xF8, // 'a'
00072     0xC1,0x04,0x1C,0x49,0x24,0xBC, // 'b'
00073     0x64,0xA1,0x07,0x00, // 'c'
00074     0x30,0x41,0x1C,0x92,0x49,0x1E, // 'd'
00075     0x64,0xBD,0x07,0x00, // 'e'
00076     0x72,0x11,0xC4,0x21,0x1C, // 'f'
00077     0x7A,0x49,0x24,0x70,0x4E,0x00, // 'g'
00078     0xC0,0x81,0x03,0xC4,0x89,0x12,0x76, // 'h'
00079     0x40,0xC4,0x44,0xE0, // 'i'
00080     0x20,0x62,0x22,0x22,0xE0, // 'j'
00081     0xC0,0x81,0x02,0xC5,0x0E,0x12,0x76, // 'k'
00082     0xC4,0x44,0x44,0x4E, // 'l'
00083     0xFF,0x12,0x44,0x91,0x24,0xED,0x80, // 'm'
00084     0xF8,0x91,0x22,0x4E,0xC0, // 'n'
00085     0x64,0xA5,0x26,0x00, // 'o'

```

```

00086     0xF1,0x24,0x92,0x71,0x0E,0x00, // 'p'
00087     0x7A,0x49,0x24,0x70,0x43,0x80, // 'q'
00088     0xF2,0x90,0x8E,0x00, // 'r'
00089     0xF4,0x18,0x2F,0x00, // 's'
00090     0x47,0x10,0x85,0x38, // 't'
00091     0xD8,0x91,0x22,0x47,0xC0, // 'u'
00092     0xD9,0x45,0x08,0x20, // 'v'
00093     0xD6,0x54,0x54,0x28,0x28, // 'w'
00094     0xD9,0x42,0x14,0xD8, // 'x'
00095     0xD9,0x45,0x08,0x20,0x8C,0x00, // 'y'
00096     0xF5,0x18,0xAF,0x00, // 'z'
00097     0x64,0x44,0x84,0x46, // '{'
00098     0xAA,0xAA,0x80, // '|'
00099     0xC4,0x44,0x24,0x4C // '}'
00100 };
00101 const GFXglyph Serif_plain_9Glyphs[] PROGMEM = {
00102 // bitmapOffset, width, height, xAdvance, xOffset, yOffset
00103     { 0, 2, 1, 4, 0, -1 }, // ' '
00104     { 1, 2, 7, 5, 1, -7 }, // '!'
00105     { 3, 4, 2, 5, 1, -7 }, // '"'
00106     { 4, 7, 6, 9, 0, -6 }, // '#'
00107     { 10, 5, 8, 7, 0, -7 }, // '$'
00108     { 15, 9, 7, 10, 0, -7 }, // '%'
00109     { 23, 8, 7, 9, 0, -7 }, // '&'
00110     { 30, 2, 2, 3, 1, -7 }, // "'"
00111     { 31, 3, 8, 5, 0, -8 }, // '('
00112     { 34, 3, 8, 5, 1, -8 }, // ')'
00113     { 37, 6, 4, 6, 0, -7 }, // '*'
00114     { 40, 6, 5, 9, 1, -5 }, // '+'
00115     { 44, 3, 2, 4, 0, -1 }, // ','
00116     { 45, 3, 1, 4, 0, -3 }, // '-'
00117     { 46, 2, 1, 4, 1, -1 }, // '.'
00118     { 47, 4, 7, 4, 0, -7 }, // '/'
00119     { 51, 6, 7, 7, 0, -7 }, // '0'
00120     { 57, 4, 7, 7, 1, -7 }, // '1'
00121     { 61, 5, 7, 7, 0, -7 }, // '2'
00122     { 66, 5, 7, 7, 0, -7 }, // '3'
00123     { 71, 6, 7, 7, 0, -7 }, // '4'
00124     { 77, 5, 7, 7, 1, -7 }, // '5'
00125     { 82, 6, 7, 7, 0, -7 }, // '6'
00126     { 88, 5, 7, 7, 1, -7 }, // '7'
00127     { 93, 6, 7, 7, 0, -7 }, // '8'
00128     { 99, 6, 7, 7, 0, -7 }, // '9'
00129     { 105, 2, 4, 4, 1, -4 }, // ':'
00130     { 106, 3, 5, 4, 0, -4 }, // ';'
00131     { 108, 7, 5, 9, 0, -5 }, // '<'
00132     { 113, 7, 3, 9, 0, -4 }, // '='
00133     { 116, 7, 5, 9, 0, -5 }, // '>'
00134     { 121, 5, 7, 6, 0, -7 }, // '?'
00135     { 126, 9, 8, 10, 0, -6 }, // '@'
00136     { 135, 8, 7, 8, 0, -7 }, // 'A'
00137     { 142, 7, 7, 8, 0, -7 }, // 'B'
00138     { 149, 7, 7, 8, 0, -7 }, // 'C'
00139     { 156, 7, 7, 8, 0, -7 }, // 'D'
00140     { 163, 7, 7, 8, 0, -7 }, // 'E'
00141     { 170, 7, 7, 7, 0, -7 }, // 'F'
00142     { 177, 7, 7, 8, 0, -7 }, // 'G'
00143     { 184, 8, 7, 9, 0, -7 }, // 'H'
00144     { 191, 4, 7, 5, 0, -7 }, // 'I'
00145     { 195, 5, 9, 5, -1, -7 }, // 'J'
00146     { 201, 8, 7, 8, 0, -7 }, // 'K'
00147     { 208, 6, 7, 7, 0, -7 }, // 'L'
00148     { 214, 10, 7, 11, 0, -7 }, // 'M'
00149     { 223, 8, 7, 9, 0, -7 }, // 'N'
00150     { 230, 7, 7, 8, 0, -7 }, // 'O'
00151     { 237, 6, 7, 7, 0, -7 }, // 'P'
00152     { 243, 7, 8, 8, 0, -7 }, // 'Q'
00153     { 250, 7, 7, 8, 0, -7 }, // 'R'
00154     { 257, 6, 7, 7, 0, -7 }, // 'S'
00155     { 263, 6, 7, 7, 0, -7 }, // 'T'
00156     { 269, 8, 7, 9, 0, -7 }, // 'U'
00157     { 276, 8, 7, 8, 0, -7 }, // 'V'
00158     { 283, 10, 7, 10, 0, -7 }, // 'W'
00159     { 292, 7, 7, 8, 0, -7 }, // 'X'
00160     { 299, 8, 7, 7, -1, -7 }, // 'Y'
00161     { 306, 6, 7, 7, 0, -7 }, // 'Z'
00162     { 312, 3, 8, 5, 0, -8 }, // '['
00163     { 315, 4, 7, 4, 0, -7 }, // '\'
00164     { 319, 3, 8, 5, 1, -8 }, // ']'
00165     { 322, 7, 2, 9, 1, -7 }, // '^'
00166     { 324, 6, 1, 6, 0, 1 }, // '_'
00167     { 325, 3, 2, 6, 1, -8 }, // '`'
00168     { 326, 6, 5, 6, 0, -5 }, // 'a'
00169     { 330, 6, 8, 7, 0, -8 }, // 'b'
00170     { 336, 5, 5, 6, 0, -5 }, // 'c'
00171     { 340, 6, 8, 6, 0, -8 }, // 'd'
00172     { 346, 5, 5, 6, 0, -5 }, // 'e'

```

```

00173     { 350, 5, 8, 4, 0, -8 }, // 'f'
00174     { 355, 6, 7, 6, 0, -5 }, // 'g'
00175     { 361, 7, 8, 7, 0, -8 }, // 'h'
00176     { 368, 4, 7, 4, 0, -7 }, // 'i'
00177     { 372, 4, 9, 4, -1, -7 }, // 'j'
00178     { 377, 7, 8, 7, 0, -8 }, // 'k'
00179     { 384, 4, 8, 4, 0, -8 }, // 'l'
00180     { 388, 10, 5, 10, 0, -5 }, // 'm'
00181     { 395, 7, 5, 7, 0, -5 }, // 'n'
00182     { 400, 5, 5, 6, 0, -5 }, // 'o'
00183     { 404, 6, 7, 7, 0, -5 }, // 'p'
00184     { 410, 6, 7, 7, 0, -5 }, // 'q'
00185     { 416, 5, 5, 5, 0, -5 }, // 'r'
00186     { 420, 5, 5, 6, 0, -5 }, // 's'
00187     { 424, 5, 6, 5, 0, -6 }, // 't'
00188     { 428, 7, 5, 7, 0, -5 }, // 'u'
00189     { 433, 6, 5, 6, 0, -5 }, // 'v'
00190     { 437, 8, 5, 9, 0, -5 }, // 'w'
00191     { 442, 6, 5, 6, 0, -5 }, // 'x'
00192     { 446, 6, 7, 6, 0, -5 }, // 'y'
00193     { 452, 5, 5, 6, 0, -5 }, // 'z'
00194     { 456, 4, 8, 7, 1, -8 }, // '{'
00195     { 460, 2, 9, 4, 1, -7 }, // '|'
00196     { 463, 4, 8, 7, 1, -8 } // '}'
00197 };
00198 const GFXfont Serif_plain_9 PROGMEM = {
00199 (uint8_t *)Serif_plain_9Bitmaps, (GFXglyph *)Serif_plain_9Glyphs, 0x20, 0x7E, 12};

```

## 2.21 globals/globals.cpp File Reference

This file contains global variables, constants, and configurations used throughout the project.

```

#include "globals.h"
#include <libraries.h>

```

### Functions

- ThreeWire **myWire** (21, 22, 4)
- IPAddress **ip** (192, 168, 1, 200)
- RtcDS1302< ThreeWire > **Rtc** (myWire)
- BfButton **btn** (BfButton::STANDALONE\_DIGITAL, 5, true, LOW)
- EthernetServer **server** (80)
- AccelStepper **motor** (AccelStepper::DRIVER, 2, 15)

### Variables

- const uint32\_t **LOOP\_TICKS** = 50 / portTICK\_PERIOD\_MS  
*Timing and Scheduling Constants.*
- const unsigned long **min\_bright\_timer** = 180000  
*Minimum time before dimming the screen (in milliseconds)*
- const unsigned long **turn\_off\_timer** = 600000  
*Timer for turning off the display (in milliseconds)*
- const int **DIR\_PIN** = 15  
*Rotary Encoder Pin Definitions.*
- const int **STEP\_PIN** = 2  
*Step pin for rotary encoder.*
- int **DT** = 16  
*Data pin for rotary encoder.*
- int **CLK** = 17

- Clock pin for rotary encoder.*
- int **lastClk** = HIGH
  - Rotary Encoder State Variables.*
- int **newClk** = digitalRead(CLK)
  - Current clock state.*
- int **dtValue** = digitalRead(DT)
  - Data pin value for rotary encoder.*
- int **counter** = 0
  - Encoder counter to track position.*
- int **angle** = 0
  - Current angle of the encoder.*
- int **aState**
  - Current state of the encoder.*
- int **aLastState**
  - Last state of the encoder.*
- long **position** = 0
  - Rotary encoder position.*
- int **lastAngle** = 0
  - Last measured angle.*
- int **centerX** = 80
  - Display Geometry Variables.*
- int **centerY** = 74
  - Y coordinate for center of display.*
- int **radius** = 40
  - Radius of UI elements.*
- const int **minBrightness** = 200
  - Brightness Control Variables.*
- const int **maxBrightness** = 230
  - Maximum brightness level.*
- const int **numBrightnessLevels** = 10
  - Total number of brightness levels.*
- int **brightnessLevel** = 5
  - Current brightness level.*
- **MenuState** **currentMenu** = INITIAL\_SCREEN
  - MenuState enum values represent different menu screens.*
- **MenuState** **previousMenu** = MAIN\_MENU
  - Previous menu state.*
- int **menuIndex** = 0
  - Menu Structure and Indexes.*
- const int **menuItems** = 4
  - Total number of main menu items.*
- String **menu** [**menuItems**] = {"MAIN MENU", "Settings", "TCP server", "Stepper motor"}
  - Main menu labels.*
- const int **submenuItems** = 4
  - Total number of submenu items.*
- String **submenu** [**submenuItems**] = {"SETTINGS", "Brightness", "Colors", "Font type"}
  - Submenu labels.*
- const int **colormenuItems** = 5
  - Number of items in color menu.*
- String **colormenu** [**colormenuItems**] = {"COLORS", "Background", "Selection bar", "Selection border", "Text"}
  - Color menu labels.*

- const int **ethernetmenultems** = 5
- String **ethernetmenu** [ethernetmenultems] = {"IP", "Port", "Subnet", "Status", "Back"}
- const int **steppermenultems** = 3
- String **steppermenu** [steppermenultems] = {"STEPPER", "Automatic", "Manual"}
- const int **manualmenultems** = 3
- String **manualmenu** [manualmenultems] = {"MANUAL", "Forward", "Backward"}
- const int **automenultems** = 1
- String **automenu** [automenultems] = {"AUTO"}
- byte **mac** [] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED }
- Ethernet Configuration.*
- char **ipchar** [16]
- char **subnetchar** [16]
- IPAddress **subnet** = Ethernet.subnetMask()
- char **ipadress** = sprintf(ipchar, "%d.%d.%d.%d", ip[0], ip[1], ip[2], ip[3])
- char **subnetadress** = sprintf(subnetchar, "%d.%d.%d.%d", subnet[0], subnet[1], subnet[2], subnet[3])
- String **ipip** = ipchar
- String **subnetsubnet** = subnetchar
- String **ethernetInfo** [5] = {ipip, "80", subnetsubnet, ""}
- Ethernet settings: IP, Port, Subnet, Status.*
- const unsigned short \* **images\_main** [] = {NULL, settings\_icon, ethernet\_icon, step\_icon}
- Icons for Menus.*
- const unsigned short \* **images\_settings** [] = {NULL, brightness\_icon, color\_icon, font\_icon}
- Settings menu icons.*
- const unsigned short \* **images\_stepper** [] = {NULL, automatic\_icon, manual\_icon}
- Stepper motor menu icons.*
- bool **screenOn** = true
- Screen Control Variables.*
- bool **engineeringMode** = false
- Flag for enabling engineering/debug mode.*
- volatile int **encoderState** = 0
- Encoder state for handling inputs.*
- RtcDateTime **lastDisplayedTime**
- Time last displayed on the screen.*
- QueueHandle\_t **screenUpdateQueue**
- Queues and Synchronization.*
- xSemaphoreHandle **gatekeeper** = 0
- Semaphore for managing screen updates.*
- const uint16\_t **backgroundpossibleColors** [] = {MINT\_GREEN, TFT\_WHITE, LIGHT\_BLUE, PALE\_GRAY, PALE\_YELLOW, PALE\_PINK, BEIGE, LAVENDER, SKY\_BLUE}
- Color Arrays for Background, Selection Bar, Border, and Text.*
- const uint16\_t **selectbarpossibleColors** [] = {TFT\_RED, TFT\_BLUE, TFT\_YELLOW, TFT\_PINK, TFT\_ORANGE, TFT\_CYAN, TFT\_SKYBLUE, TFT\_VIOLET}
- Possible selection bar colors.*
- const uint16\_t **borderpossibleColors** [] = {TFT\_BLACK, TFT\_RED, TFT\_GREENYELLOW, TFT\_BLUE, TFT\_YELLOW, TFT\_PINK, TFT\_ORANGE, TFT\_CYAN, TFT\_LIGHTGREY, TFT\_SKYBLUE, TFT\_VIOLET}
- Possible border colors.*
- const uint16\_t **textpossibleColors** [] = {TFT\_BLACK, DARK\_ORANGE, DARK\_BLUE, DARK\_GREEN}
- Possible text colors.*
- const int **backgroundcolors** = 9
- Color Count Variables.*
- const int **selectbarcolors** = 8
- Number of selection bar colors.*

- const int **textcolors** = 4  
*Number of text colors.*
- const int **bordercolors** = 11  
*Number of border colors.*
- const int **selectnumcolors** [] = {backgroundcolors,selectbarcolors,textcolors,bordercolors}
- MenuState colormenuArray [] = {MenuState::SELECTION\_BAR, MenuState::BACKGROUND, MenuState::TEXT, MenuState::SELECTION\_BORDER}  
*Color Menu Arrays.*
- String **background** [backgroundcolors] = {"Mint","White","Light blue", "Pale gray", "Yellow","Pale pink","Beige", "Lavender","Skyblue"}  
*Background color labels.*
- String **selectionbar** [selectbarcolors] = {"Red", "Blue", "Yellow","Pink", "Orange","Cyan", "Skyblue","Violet"}  
*Selection bar color labels.*
- String **border** [bordercolors] = {"Black","Red","Green","Blue","Yellow","Pink","Orange","Cyan","Light grey","Skyblue","Violet"}  
*Border color labels.*
- String **text** [textcolors] = {"Black","Orange","Dark blue","Dark green"}  
*Text color labels.*
- int **selectedBackgroundColor** = 1  
*Selected Colors.*
- int **selectedTextColor** = 0  
*Selected text color index.*
- int **selectedBarColor** = 5  
*Selected selection bar color index.*
- int **selectedBorderColor** = 0  
*Selected border color index.*
- int \* **selection** [] = {NULL, &selectedBackgroundColor, &selectedBarColor, &selectedBorderColor, &selectedTextColor}  
*Array for selected colors.*
- const GFXfont \* **possibleFont12** [] = {NULL, sansserif12, mono12, arimo12, serif12}  
*Font Configuration.*
- const GFXfont \* **possibleFont9** [] = {NULL, sansserif9, mono9, arimo9, serif9}  
*Array of possible fonts (9-point size)*
- const int **numFonts** = sizeof(possibleFont12) / sizeof(possibleFont12[0])  
*Number of font options.*
- String **fontType** [numFonts] = {"TEXT FONT", "SansSerif", "Monospaced", "Arimo", "Serif"}  
*Font type labels.*
- const GFXfont \* **textType12** = possibleFont12[4]  
*Selected 12-point font.*
- const GFXfont \* **textType9** = possibleFont9[4]  
*Selected 9-point font.*
- TFT\_eSPI **tft** = TFT\_eSPI()  
*Hardware Initialization.*
- TickType\_t **previousTick** = 0
- TickType\_t **previousTime** = 0

### 2.21.1 Detailed Description

This file contains global variables, constants, and configurations used throughout the project.

#### Author

Juan Alberto Serrano Redondo.

## 2.21.2 Variable Documentation

### 2.21.2.1 backgroundcolors

```
const int backgroundcolors = 9
```

Color Count Variables.

Number of background colors.

Number of background colors

### 2.21.2.2 backgroundpossibleColors

```
const uint16_t backgroundpossibleColors[] = {MINT_GREEN, TFT_WHITE, LIGHT_BLUE, PALE_GRAY,  
PALE_YELLOW, PALE_PINK, BEIGE, LAVENDER, SKY_BLUE}
```

Color Arrays for Background, Selection Bar, Border, and Text.

Possible background colors.

Possible background colors

### 2.21.2.3 centerX

```
int centerX = 80
```

Display Geometry Variables.

X coordinate for center of display.

X coordinate for center of display

### 2.21.2.4 colormenuArray

```
MenuState colormenuArray[] = {MenuState::SELECTION_BAR, MenuState::BACKGROUND, MenuState::TEXT,  
MenuState::SELECTION_BORDER}
```

Color Menu Arrays.

Color menu states.

Color menu states

### 2.21.2.5 currentMenu

```
MenuState currentMenu = INITIAL_SCREEN
```

MenuState enum values represent different menu screens.

Current menu state.

Current menu state

#### 2.21.2.6 DIR\_PIN

```
const int DIR_PIN = 15
```

Rotary Encoder Pin Definitions.

Direction pin for rotary encoder.

Direction pin for rotary encoder

#### 2.21.2.7 images\_main

```
const unsigned short* images_main[] = {NULL, settings_icon, ethernet_icon, step_icon}
```

Icons for Menus.

Main menu icons.

Main menu icons

#### 2.21.2.8 lastClk

```
int lastClk = HIGH
```

Rotary Encoder State Variables.

Last clock state.

Last clock state

#### 2.21.2.9 LOOP\_TICKS

```
const uint32_t LOOP_TICKS = 50 / portTICK_PERIOD_MS
```

Timing and Scheduling Constants.

Loop period in ticks for RTOS task.

Loop period in ticks for RTOS task

#### 2.21.2.10 menuIndex

```
int menuIndex = 0
```

Menu Structure and Indexes.

Index for current menu item.

Index for current menu item



### 2.21.2.11 minBrightness

```
const int minBrightness = 200
```

Brightness Control Variables.

Minimum brightness level.

Minimum brightness level

### 2.21.2.12 possibleFont12

```
const GFXfont* possibleFont12[] = {NULL, sansserif12, mono12, arimo12, serif12}
```

Font Configuration.

Array of possible fonts (12-point size)

### 2.21.2.13 screenOn

```
bool screenOn = true
```

Screen Control Variables.

Flag indicating if the screen is currently on.

Flag indicating if the screen is currently on

### 2.21.2.14 screenUpdateQueue

```
QueueHandle_t screenUpdateQueue
```

Queues and Synchronization.

Queue for handling screen update commands.

Queue for handling screen update commands

### 2.21.2.15 selectedBackgroundColor

```
int selectedBackgroundColor = 1
```

Selected Colors.

Selected background color index.

Selected background color index

## 2.22 globals/globals.h File Reference

Global variable and constant definitions for the project.

```
#include <libraries.h>
```

### Macros

- **#define LOAD\_GFXFF**
- **#define BRIGHTNESS\_PIN 25**  
*Pin for brightness control.*
- **#define LIGHT\_GRAY 0xC618**  
*Light gray color for UI elements.*
- **#define LIGHT\_BLUE 0xE7DF**  
*Light blue color for UI elements.*
- **#define PALE\_GRAY 0xE71C**  
*Pale gray color for UI elements.*
- **#define MINT\_GREEN 0x97F6**  
*Mint green color for UI elements.*
- **#define PALE\_YELLOW 0xFFFFB**  
*Pale yellow color for UI elements.*
- **#define PALE\_PINK 0xFDBC**  
*Pale pink color for UI elements.*
- **#define BEIGE 0xFF7B**  
*Beige color for UI elements.*
- **#define LAVENDER 0xE5FF**  
*Lavender color for UI elements.*
- **#define SKY\_BLUE 0x9FFF**  
*Sky blue color for UI elements.*
- **#define DARK\_ORANGE 0xDBC9**  
*Dark orange color for UI elements.*
- **#define DARK\_BLUE 0x00B1**  
*Dark blue color for UI elements.*
- **#define DARK\_GREEN 0x22A2**  
*Dark green color for UI elements.*

### Enumerations

- enum **MenuState** {  
[MAIN\\_MENU](#) , [SETTINGS](#) , [BRIGHTNESS](#) , [COLOR](#) ,  
[TEXT\\_TYPE](#) , [BACKGROUND](#) , [SELECTION\\_BAR](#) , [SELECTION\\_BORDER](#) ,  
[TEXT](#) , [SERVIDOR\\_TCP](#) , [STEPPER\\_MOTOR](#) , [MANUAL](#) ,  
[AUTO](#) , [INITIAL\\_SCREEN](#) }  
*Defines various states of the menu interface.*
- enum **ScreenUpdateCommand** { [UPDATE\\_MAIN\\_MENU](#) , [UPDATE\\_CONFIG\\_BRIGHTNESS](#) , [UPDATE\\_CONFIG\\_BACKGROUND](#) , [UPDATE\\_CONFIG\\_ASPECT](#) }  
*Enum for screen update commands.*

## Variables

- const uint32\_t [LOOP\\_TICKS](#)  
*Loop period in ticks for RTOS task.*
- const unsigned long **min\_bright\_timer**  
*Minimum time before dimming the screen (in milliseconds)*
- const unsigned long **turn\_off\_timer**  
*Timer for turning off the display (in milliseconds)*
- const int [DIR\\_PIN](#)  
*Direction pin for rotary encoder.*
- const int **STEP\_PIN**  
*Step pin for rotary encoder.*
- int **DT**  
*Data pin for rotary encoder.*
- int **CLK**  
*Clock pin for rotary encoder.*
- int [lastClk](#)  
*Last clock state.*
- int **newClk**  
*Current clock state.*
- int **dtValue**  
*Data pin value for rotary encoder.*
- int **counter**  
*Encoder counter to track position.*
- int **angle**  
*Current angle of the encoder.*
- int **aState**  
*Current state of the encoder.*
- int **aLastState**  
*Last state of the encoder.*
- long **position**  
*Rotary encoder position.*
- int **lastAngle**  
*Last measured angle.*
- int [centerX](#)  
*X coordinate for center of display.*
- int **centerY**  
*Y coordinate for center of display.*
- int **radius**  
*Radius of UI elements.*
- const int [minBrightness](#)  
*Minimum brightness level.*
- const int **maxBrightness**  
*Maximum brightness level.*
- const int **numBrightnessLevels**  
*Total number of brightness levels.*
- int **brightnessLevel**  
*Current brightness level.*
- [MenuState](#) **currentMenu**  
*Current menu state.*
- [MenuState](#) **previousMenu**

- Previous menu state.*
- int [menuIndex](#)
  - Index for current menu item.*
- const int **menuItems**
  - Total number of main menu items.*
- String **menu** []
  - Main menu labels.*
- const int **submenuItems**
  - Total number of submenu items.*
- String **submenu** []
  - Submenu labels.*
- const int **colormenuItems**
  - Number of items in color menu.*
- String **colormenu** []
  - Color menu labels.*
- const int **ethernetmenuItems**
- String **ethernetmenu** []
- const int **steppermenuItems**
- String **steppermenu** []
- const int **manualmenuItems**
- String **manualmenu** []
- const int **automenuItems**
- String **automenu** []
- byte **mac** []
  - Ethernet Configuration.*
- IPAddress **ip**
- EthernetServer **server**
- char **ipchar** [16]
- char **subnetchar** [16]
- IPAddress **subnet**
- char **ipadress**
- char **subnetadress**
- String **ipip**
- String **subnetsubnet**
- String **ethernetInfo** [5]
  - Ethernet settings: IP, Port, Subnet, Status.*
- const unsigned short \* [images\\_main](#) []
  - Main menu icons.*
- const unsigned short \* **images\_settings** []
  - Settings menu icons.*
- const unsigned short \* **images\_stepper** []
  - Stepper motor menu icons.*
- bool [screenOn](#)
  - Flag indicating if the screen is currently on.*
- bool **engineeringMode**
  - Flag for enabling engineering/debug mode.*
- volatile int **encoderState**
  - Encoder state for handling inputs.*
- RtcDateTime **lastDisplayedTime**
  - Time last displayed on the screen.*
- QueueHandle\_t [screenUpdateQueue](#)
  - Queue for handling screen update commands.*

- xSemaphoreHandle **gatekeeper**  
*Semaphore for managing screen updates.*
- const uint16\_t **backgroundpossibleColors** []  
*Possible background colors.*
- const uint16\_t **selectbarpossibleColors** []  
*Possible selection bar colors.*
- const uint16\_t **borderpossibleColors** []  
*Possible border colors.*
- const uint16\_t **textpossibleColors** []  
*Possible text colors.*
- const int **backgroundcolors**  
*Number of background colors.*
- const int **selectbarcolors**  
*Number of selection bar colors.*
- const int **textcolors**  
*Number of text colors.*
- const int **bordercolors**  
*Number of border colors.*
- const int **selectnumcolors** []
- MenuState **colormenuArray** []  
*Color menu states.*
- String **background** []  
*Background color labels.*
- String **selectionbar** []  
*Selection bar color labels.*
- String **border** []  
*Border color labels.*
- String **text** []  
*Text color labels.*
- int **selectedBackgroundColor**  
*Selected background color index.*
- int **selectedTextColor**  
*Selected text color index.*
- int **selectedBarColor**  
*Selected selection bar color index.*
- int **selectedBorderColor**  
*Selected border color index.*
- int \* **selection** []  
*Array for selected colors.*
- const GFXfont \* **possibleFont12** []  
*Array of possible fonts (12-point size)*
- const GFXfont \* **possibleFont9** []  
*Array of possible fonts (9-point size)*
- const int **numFonts**  
*Number of font options.*
- String **fontType** []  
*Font type labels.*
- const GFXfont \* **textType12**  
*Selected 12-point font.*
- const GFXfont \* **textType9**  
*Selected 9-point font.*

- BfButton **btn**
- ThreeWire **myWire**
- RtcDS1302< ThreeWire > **Rtc**
- AccelStepper **motor**
- TFT\_eSPI **tft**

*Hardware Initialization.*

- TickType\_t **previousTick**
- TickType\_t **previousTime**

### 2.22.1 Detailed Description

Global variable and constant definitions for the project.

This header file contains the global definitions for UI colors, menu states, rotary encoder configurations, display settings, and other system-wide constants and variables. It also declares external variables and enums that are used across different modules of the program, ensuring centralized configuration and ease of modification.

The file includes:

- Custom color definitions for user interface (UI) elements.
- Pin assignments for hardware components (brightness control, rotary encoder).
- Timing constants and menu states (main menu, settings, motor control).
- Rotary encoder state tracking and UI geometry settings.
- Brightness control and menu structure.
- Ethernet configuration parameters and TCP server settings.
- Screen control and synchronization using FreeRTOS tasks and queues.
- Enum definitions for menus and screen updates.

#### Author

Juan Alberto Serrano Redondo.

### 2.22.2 Enumeration Type Documentation

#### 2.22.2.1 MenuState

enum `MenuState`

Defines various states of the menu interface.

#### Enumerator

MAIN_MENU	Main menu.
SETTINGS	Settings menu.
BRIGHTNESS	Brightness control menu.
COLOR	Color configuration menu.
TEXT_TYPE	Text font selection.

## Enumerator

BACKGROUND	Background color selection.
SELECTION_BAR	Selection bar color.
SELECTION_BORDER	Selection border color.
TEXT	Text color configuration.
SERVIDOR_TCP	TCP server configuration.
STEPPER_MOTOR	Stepper motor control.
MANUAL	Manual stepper control.
AUTO	Automatic stepper control.
INITIAL_SCREEN	Initial splash screen.

## 2.22.2.2 ScreenUpdateCommand

```
enum ScreenUpdateCommand
```

Enum for screen update commands.

## Enumerator

UPDATE_MAIN_MENU	Command to update the main menu.
UPDATE_CONFIG_BRIGHTNESS	Command to update brightness configuration.
UPDATE_CONFIG_BACKGROUND_COLOR	Command to update background color.
UPDATE_CONFIG_ASPECT	Command to update UI aspects.

## 2.22.3 Variable Documentation

## 2.22.3.1 backgroundcolors

```
const int backgroundcolors [extern]
```

Number of background colors.

Number of background colors.

Number of background colors

## 2.22.3.2 backgroundpossibleColors

```
const uint16_t backgroundpossibleColors[] [extern]
```

Possible background colors.

Possible background colors.

Possible background colors

### 2.22.3.3 centerX

```
int centerX [extern]
```

X coordinate for center of display.

X coordinate for center of display.

X coordinate for center of display

### 2.22.3.4 colormenuArray

```
MenuState colormenuArray[] [extern]
```

Color menu states.

Color menu states.

Color menu states

### 2.22.3.5 currentMenu

```
MenuState currentMenu [extern]
```

Current menu state.

Current menu state.

Current menu state

### 2.22.3.6 DIR\_PIN

```
const int DIR_PIN [extern]
```

Direction pin for rotary encoder.

Direction pin for rotary encoder.

Direction pin for rotary encoder

### 2.22.3.7 images\_main

```
const unsigned short* images_main[] [extern]
```

Main menu icons.

Main menu icons.

Main menu icons



### 2.22.3.8 lastClk

```
int lastClk [extern]
```

Last clock state.

Last clock state.

Last clock state

### 2.22.3.9 LOOP\_TICKS

```
const uint32_t LOOP_TICKS [extern]
```

Loop period in ticks for RTOS task.

Loop period in ticks for RTOS task.

Loop period in ticks for RTOS task

### 2.22.3.10 menuIndex

```
int menuIndex [extern]
```

Index for current menu item.

Index for current menu item.

Index for current menu item

### 2.22.3.11 minBrightness

```
const int minBrightness [extern]
```

Minimum brightness level.

Minimum brightness level.

Minimum brightness level

### 2.22.3.12 possibleFont12

```
const GFXfont* possibleFont12[] [extern]
```

Array of possible fonts (12-point size)

Array of possible fonts (12-point size)

### 2.22.3.13 screenOn

```
bool screenOn [extern]
```

Flag indicating if the screen is currently on.

Flag indicating if the screen is currently on.

Flag indicating if the screen is currently on

### 2.22.3.14 screenUpdateQueue

```
QueueHandle_t screenUpdateQueue [extern]
```

Queue for handling screen update commands.

Queue for handling screen update commands.

Queue for handling screen update commands

### 2.22.3.15 selectedBackgroundColor

```
int selectedBackgroundColor [extern]
```

Selected background color index.

Selected background color index.

Selected background color index

## 2.23 globals.h

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

```
00001
00023 #ifndef GLOBALS_H
00024 #define GLOBALS_H
00025
00026 #include <libraries.h>
00027
00028
00029 // Graphics and Pin Definitions
00030 #define LOAD_GFXFF
00031 #define BRIGHTNESS_PIN 25
00032
00033 // Custom Color Definitions
00034 #define LIGHT_GRAY 0xC618
00035 #define LIGHT_BLUE 0xE7DF
00036 #define PALE_GRAY 0xE71C
00037 #define MINT_GREEN 0x97F6
00038 #define PALE_YELLOW 0xFFFB
00039 #define PALE_PINK 0xFDBC
00040 #define BEIGE 0xFF7B
00041 #define LAVENDER 0xE5FF
00042 #define SKY_BLUE 0x9FFF
00043 #define DARK_ORANGE 0xDBC9
00044 #define DARK_BLUE 0x00B1
00045 #define DARK_GREEN 0x22A2
00046
00047 // Timing and Scheduling Constants
00048 extern const uint32_t LOOP_TICKS;
00049 extern const unsigned long min_bright_timer;
00050 extern const unsigned long turn_off_timer;
```

```

00051
00052 // Rotary Encoder Pin Definitions
00053 extern const int DIR_PIN;
00054 extern const int STEP_PIN;
00055 extern int DT;
00056 extern int CLK;
00057
00058 // Rotary Encoder State Variables
00059 extern int lastClk;
00060 extern int newClk;
00061 extern int dtValue;
00062 extern int counter;
00063 extern int angle;
00064 extern int aState;
00065 extern int aLastState;
00066 extern long position;
00067 extern int lastAngle;
00068
00069 // Display Geometry Variables
00070 extern int centerX;
00071 extern int centerY;
00072 extern int radius;
00073
00074 // Brightness Control Variables
00075 extern const int minBrightness;
00076 extern const int maxBrightness;
00077 extern const int numBrightnessLevels;
00078 extern int brightnessLevel;
00079
00080 // Enum for Menu States
00081 enum MenuState {
00082     MAIN_MENU,
00083     SETTINGS,
00084     BRIGHTNESS,
00085     COLOR,
00086     TEXT_TYPE,
00087     BACKGROUND,
00088     SELECTION_BAR,
00089     SELECTION_BORDER,
00090     TEXT,
00091     SERVIDOR_TCP,
00092     STEPPER_MOTOR,
00093     MANUAL,
00094     AUTO,
00095     INITIAL_SCREEN
00096 };
00097
00098 extern MenuState currentMenu;
00099 extern MenuState previousMenu;
00100
00101 // Menu Structure and Indexes
00102 extern int menuIndex;
00103 extern const int menuItems;
00104 extern String menu[];
00105 extern const int submenuItems;
00106 extern String submenu[];
00107 extern const int colormenuItems;
00108 extern String colormenu[];
00109 extern const int ethernetmenuItems;
00110 extern String ethernetmenu[];
00111 extern const int steppermenuItems;
00112 extern String steppermenu[];
00113 extern const int manualmenuItems;
00114 extern String manualmenu[];
00115 extern const int automenuItems;
00116 extern String automenu[];
00117 // Ethernet Configuration
00118 extern byte mac[];
00119 extern IPAddress ip;
00120 extern EthernetServer server;
00121 extern char ipchar[16];
00122 extern char subnetchar[16];
00123 extern IPAddress subnet;
00124 extern char ipaddress;
00125 extern char subnetaddress;
00126 extern String ipip;
00127 extern String subnetsubnet;
00128 extern String ethernetInfo[5];
00129
00130 // Icons for Menus
00131 extern const unsigned short *images_main[];
00132 extern const unsigned short *images_settings[];
00133 extern const unsigned short *images_stepper[];
00134
00135 // Screen Control Variables
00136 extern bool screenOn;

```

```

00142 extern bool engineeringMode;
00143 extern volatile int encoderState;
00144 extern RtcDateTime lastDisplayedTime;
00145
00146 // Queues and Synchronization
00147 extern QueueHandle_t screenUpdateQueue;
00148 extern xSemaphoreHandle gatekeeper;
00149
00154 enum ScreenUpdateCommand {
00155     UPDATE_MAIN_MENU,
00156     UPDATE_CONFIG_BRIGHTNESS,
00157     UPDATE_CONFIG_BACKGROUND_COLOR,
00158     UPDATE_CONFIG_ASPECT
00159 };
00160
00161 // Color Arrays for Background, Selection Bar, Border, and Text
00162 extern const uint16_t backgroundpossibleColors[];
00163 extern const uint16_t selectbarpossibleColors[];
00164 extern const uint16_t borderpossibleColors[];
00165 extern const uint16_t textpossibleColors[];
00166
00167 // Color Count Variables
00168 extern const int backgroundcolors;
00169 extern const int selectbarcolors;
00170 extern const int textcolors;
00171 extern const int bordercolors;
00172 extern const int selectnumcolors[];
00173
00174 // Color Menu Arrays
00175 extern MenuState colormenuArray[];
00176 extern String background[];
00177 extern String selectionbar[];
00178 extern String border[];
00179 extern String text[];
00180
00181 // Selected Colors
00182 extern int selectedBackgroundColor;
00183 extern int selectedTextColor;
00184 extern int selectedBarColor;
00185 extern int selectedBorderColor;
00186 extern int *selection[];
00187
00188 // Font Configuration
00189 extern const GFXfont* possibleFont12[];
00190 extern const GFXfont* possibleFont9[];
00191 extern const int numFonts;
00192 extern String fontType[];
00193 extern const GFXfont* textType12;
00194 extern const GFXfont* textType9;
00195
00196 extern BfButton btn;
00197 extern ThreeWire myWire;
00198 extern RtcDS1302<ThreeWire> Rtc;
00199 extern AccelStepper motor;
00200 extern TFT_eSPI tft;
00201 extern TickType_t previousTick;
00202 extern TickType_t previousTime;
00203
00204
00205 #endif // GLOBALS_H

```

## 2.24 GUI/GUI.cpp File Reference

Implementation of the functions related to the graphical user interface (GUI).

```
#include "GUI.h"
```

### Functions

- void `printDateTime` (const RtcDateTime &dt)  
*Displays the date and time on the screen and in the serial monitor.*
- void `drawNeedle` (int angle, int length, uint32\_t color)  
*Draws a needle on the dial.*

- void `drawDialMarks` ()  
*Draws the dial marks.*
- void `updateDial` ()  
*Updates the dial by moving the needle to the new position.*
- void `setBrightness` (int level)  
*Sets the screen brightness level.*
- void `drawBrightnessBar` (int level)  
*Draws the brightness bar on the screen.*
- void `drawSelectionBar` (int16\_t y, bool isSelected, int currentIndex, int totalItems)  
*Draws a selection bar on the screen.*

### 2.24.1 Detailed Description

Implementation of the functions related to the graphical user interface (GUI).

This file contains the implementation of the functions to handle the GUI, such as drawing the dial needle, selection bar, and other functionalities.

#### Author

Juan Alberto Serrano Redondo.

### 2.24.2 Function Documentation

#### 2.24.2.1 `drawBrightnessBar()`

```
void drawBrightnessBar (  
    int level)
```

Draws the brightness bar on the screen.

##### Parameters

<i>level</i>	Current brightness level.
--------------	---------------------------

#### 2.24.2.2 `drawDialMarks()`

```
void drawDialMarks ()
```

Draws the dial marks.

Draws the marks and numbers at regular intervals around the dial.

#### 2.24.2.3 `drawNeedle()`

```
void drawNeedle (  
    int angle,  
    int length,  
    uint32_t color)
```

Draws a needle on the dial.

This function draws a needle on the dial based on the specified angle and length.

## Parameters

<i>angle</i>	Angle of the needle.
<i>length</i>	Length of the needle.
<i>color</i>	Color of the needle.

**2.24.2.4 drawSelectionBar()**

```
void drawSelectionBar (
    int16_t y,
    bool isSelected,
    int currentIndex,
    int totalItems)
```

Draws a selection bar on the screen.

This function draws a selection bar at the specified vertical position. The bar will be drawn differently based on whether it is selected or not, and its appearance can change depending on the current and total items.

## Parameters

<i>y</i>	Vertical position of the bar (in pixels).
<i>isSelected</i>	A boolean flag indicating whether the bar is selected.
<i>currentIndex</i>	Index of the currently selected item (starting from 0).
<i>totalItems</i>	Total number of items in the selection list.

**2.24.2.5 printDateTime()**

```
void printDateTime (
    const RtcDateTime & dt)
```

Displays the date and time on the screen and in the serial monitor.

This function takes a RtcDateTime object, formats the time as HH:MM, and displays it both on the serial monitor and the TFT screen.

## Parameters

<i>dt</i>	The current date and time in RtcDateTime format.
-----------	--

**2.24.2.6 setBrightness()**

```
void setBrightness (
    int level)
```

Sets the screen brightness level.

## Parameters

<i>level</i>	The brightness level to set.
--------------	------------------------------

### 2.24.2.7 updateDial()

```
void updateDial ()
```

Updates the dial by moving the needle to the new position.

This function is responsible for refreshing the dial display by updating the needle's position based on new data or changes in the system state.

## 2.25 GUI/GUI.h File Reference

Graphical User Interface (GUI) module for managing the display elements.

```
#include <globals/globals.h>
#include <libraries.h>
```

## Functions

- void [drawNeedle](#) (int [angle](#), int length, uint32\_t color)  
*Draws a needle on the dial.*
- void [drawDialMarks](#) ()  
*Draws the dial marks.*
- void [drawSelectionBar](#) (int16\_t y, bool isSelected, int currentIndex, int totalItems)  
*Draws a selection bar on the screen.*
- void [updateDial](#) ()  
*Updates the dial by moving the needle to the new position.*
- void [setBrightness](#) (int level)  
*Sets the screen brightness level.*
- void [drawBrightnessBar](#) (int level)  
*Draws the brightness bar on the screen.*
- void [printDateTime](#) (const RtcDateTime &dt)  
*Displays the date and time on the screen and in the serial monitor.*

### 2.25.1 Detailed Description

Graphical User Interface (GUI) module for managing the display elements.

This header file contains the function declarations for managing the graphical user interface (GUI) of the system. The functions provided here allow drawing different UI elements like needles, dials, selection bars, and adjusting the screen's brightness. This module interacts with the hardware display and manages visual elements based on user input or system state changes.

The primary purpose of this module is to offer functions that facilitate displaying information in a structured and visually intuitive manner on the screen, including dynamic updates such as moving the needle on a dial, changing brightness levels, and displaying the date and time.

It is assumed that the GUI interacts with a TFT display and uses a rotary encoder for user input, as well as other UI elements such as selection bars for menu navigation.

The file also defines various graphical components like brightness bars, dials, and selection bars.

Dependencies:

- This file relies on the global variables defined in [globals.h](#) and the libraries included in [libraries.h](#).

#### Author

Juan Alberto Serrano Redondo.

### 2.25.2 Function Documentation

#### 2.25.2.1 drawBrightnessBar()

```
void drawBrightnessBar (  
    int level)
```

Draws the brightness bar on the screen.

This function draws a visual representation of the current brightness level on the screen, showing the user the current setting.

#### Parameters

<i>level</i>	Current brightness level.
<i>level</i>	Current brightness level.

#### 2.25.2.2 drawDialMarks()

```
void drawDialMarks ()
```

Draws the dial marks.

This function draws the marks around the dial, including numbers at angles 0, 60, 120, etc. It is used to set up the dial's appearance with evenly spaced markings.

Draws the marks and numbers at regular intervals around the dial.



### 2.25.2.3 drawNeedle()

```
void drawNeedle (  
    int angle,  
    int length,  
    uint32_t color)
```

Draws a needle on the dial.

This function draws a needle on the dial at the specified angle and length, using the given color.

**Parameters**

<i>angle</i>	The angle at which to draw the needle (in degrees).
<i>length</i>	The length of the needle.
<i>color</i>	The color of the needle, represented as a 32-bit unsigned integer (usually RGB color).

This function draws a needle on the dial based on the specified angle and length.

**Parameters**

<i>angle</i>	Angle of the needle.
<i>length</i>	Length of the needle.
<i>color</i>	Color of the needle.

**2.25.2.4 drawSelectionBar()**

```
void drawSelectionBar (
    int16_t y,
    bool isSelected,
    int currentIndex,
    int totalItems)
```

Draws a selection bar on the screen.

This function draws a selection bar at the specified vertical position. The bar will be drawn differently based on whether it is selected or not, and its appearance can change depending on the current and total items.

**Parameters**

<i>y</i>	Vertical position of the bar (in pixels).
<i>isSelected</i>	A boolean flag indicating whether the bar is selected.
<i>currentIndex</i>	Index of the currently selected item (starting from 0).
<i>totalItems</i>	Total number of items in the selection list.

**2.25.2.5 printDateTime()**

```
void printDateTime (
    const RtcDateTime & dt)
```

Displays the date and time on the screen and in the serial monitor.

This function prints the current date and time on the screen, as well as in the serial monitor for debugging purposes.

**Parameters**

<i>dt</i>	The current date and time in RtcDateTime format, typically retrieved from a real-time clock (RTC).
-----------	--

This function takes a RtcDateTime object, formats the time as HH:MM, and displays it both on the serial monitor and the TFT screen.

## Parameters

<i>dt</i>	The current date and time in RtcDateTime format.
-----------	--

**2.25.2.6 setBrightness()**

```
void setBrightness (
    int level)
```

Sets the screen brightness level.

This function adjusts the brightness level of the screen based on the provided level value.

## Parameters

<i>level</i>	The brightness level to set (usually between a predefined min and max value).
<i>level</i>	The brightness level to set.

**2.25.2.7 updateDial()**

```
void updateDial ()
```

Updates the dial by moving the needle to the new position.

This function is responsible for refreshing the dial display by updating the needle's position based on new data or changes in the system state.

**2.26 GUI.h**

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

```
00001 #ifndef GUI_H
00002 #define GUI_H
00003
00004 #include <globals/globals.h>
00005 #include <libraries.h>
00006
00040 void drawNeedle(int angle, int length, uint32_t color);
00041
00048 void drawDialMarks();
00049
00062 void drawSelectionBar(int16_t y, bool isSelected, int currentIndex, int totalItems);
00063
00070 void updateDial();
00071
00079 void setBrightness(int level);
00080
00089 void drawBrightnessBar(int level);
00090
00098 void printDateTime(const RtcDateTime& dt);
00099
00100 #endif // GUI_H
```

## 2.27 automatic\_icon.h

```

00001 #if defined(__AVR__)
00002     #include <avr/pgmspace.h>
00003 #elif defined(__PIC32MX__)
00004     #define PROGMEM
00005 #elif defined(__arm__)
00006     #define PROGMEM
00007 #endif
00008
00009 const unsigned short automatic_icon[ ] PROGMEM={0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00010     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00011     0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00012     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00013     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00014     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00015     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00016     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00017     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00018     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00019     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00020     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00021     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00022     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00023     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00024     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00025     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00026     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00027     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00028     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00029     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00030     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00031     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00032     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00033     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00034     0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,

```

## 2.28 brightness\_icon.h

Generated by Doxygen



```

00022 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00023 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0x0000, 0xffff, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00024 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
      0xffff, 0x0000, 0xffff, 0xffff, 0x0000, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00025 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
      0x0000, 0xffff, 0xffff, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00026 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00027 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00028 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0x0000, 0x0000, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00029 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0x0000, 0x0000, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00030 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00031 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00032 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00033 0xffff, 0xffff, 0xffdf, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00034 0xffff, 0xffff, 0xffff, 0xf7be, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0xf7be, 0xffff, 0xffff,
00035 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0xffff, 0xffff, 0xffff,
00036 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffdf, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0xffdf, 0xffff, 0xffff, 0xffff,
00037 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffdf, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffdf,
      0xffff, 0xffff, 0xffff,
00038 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffdf, 0xffdf, 0xffff, 0xffff, 0xf7be,
      0xf7d, 0xf7d, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffdf, 0xffdf, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff);

```

## 2.30 empty\_sun.h

```

00001 // Generated by      : ImageConverter 565 Online
00002 // Generated from    : luna.jpg
00003 // Time generated    : Sun, 07 Jul 24 20:42:29 +0200 (Server timezone: CET)
00004 // Image Size       : 50x49 pixels
00005 // Memory usage      : 4900 bytes
00006
00007
00008 #if defined(__AVR__)
00009     #include <avr/pgmspace.h>
00010 #elif defined(__PIC32MX__)
00011     #define PROGMEM
00012 #elif defined(__arm__)
00013     #define PROGMEM
00014 #endif
00015
00016 const unsigned short empty_sun[2450] PROGMEM={
00017 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00018 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00019 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00020 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,

```





```

00003 // Time generated : Sun, 07 Jul 24 20:42:29 +0200 (Server timezone: CET)
00004 // Image Size      : 50x49 pixels
00005 // Memory usage     : 4900 bytes
00006
00007
00008 #if defined(__AVR__)
00009     #include <avr/pgmspace.h>
00010 #elif defined(__PIC32MX__)
00011     #define PROGMEM
00012 #elif defined(__arm__)
00013     #define PROGMEM
00014 #endif
00015
00016 const unsigned short ethernet_active[2450] PROGMEM={0x0000, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00017 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000,
00018 0x0000, 0xffff, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000,
00019 0x0000, 0xffff, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0x0000,
00020 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0xffff, 0x0000,
00021 0x0000, 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000,
00022 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000,
00023 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000,
00024 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000,
00025 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff,
00026 0x0000, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0x0000,
0x0000,
0x0000, 0x0000, 0x0000,
00027 0x0000, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0x0000,
0x0000,
0x0000, 0x0000, 0x0000,
00028 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff,
0xffff, 0x0000,
00029 0x0000, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000,
0x0000, 0x0000, 0x0000,
00030 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000,
0x0000, 0x0000, 0x0000};

```

## 2.32 ethernet\_icon.h

```

00001 // Generated by      : ImageConverter 565 Online
00002 // Generated from     : luna.jpg
00003 // Time generated    : Sun, 07 Jul 24 20:42:29 +0200 (Server timezone: CET)
00004 // Image Size        : 50x49 pixels
00005 // Memory usage      : 4900 bytes
00006
00007
00008 #if defined(__AVR__)
00009     #include <avr/pgmspace.h>
00010 #elif defined(__PIC32MX__)
00011     #define PROGMEM
00012 #elif defined(__arm__)
00013     #define PROGMEM
00014 #endif
00015
00016 const unsigned short ethernet_icon[] PROGMEM={0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000,
00017 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000,
0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
0xffff,
0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00018 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0xffff, 0xffff,
00019 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
0x0000,
0x0000, 0xffff, 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000,
0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00020 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
0x0000,
0x0000, 0xffff, 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000,
0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff,
00021 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
0x0000,
0x0000, 0xffff, 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000,
0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
00022 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
0x0000,
0x0000, 0xffff, 0x0000, 0x0000, 0xffff, 0x0000, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,

```



Generated by Doxygen

## 2.34 full\_sun.h

```
00001 // Generated by      : ImageConverter 565 Online
00002 // Generated from    : sol.jpg
00003 // Time generated   : Sun, 07 Jul 24 20:42:51 +0200   (Server timezone: CET)
00004 // Image Size      : 50x49 pixels
00005 // Memory usage    : 4900 bytes
00006
00007
00008 #if defined(__AVR__)
00009     #include <avr/pgmspace.h>
00010 #elif defined(__PIC32MX__)
00011     #define PROGMEM
00012 #elif defined(__arm__)
00013     #define PROGMEM
00014 #endif
00015
00016 const unsigned short full_sun[2450] PROGMEM={0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00017     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00018     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00019     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00020     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00021     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00022     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00023     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00024     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00025     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00026     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00027     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00028     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00029     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00030     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00031     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00032     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00033     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00034     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00035     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00036     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00037     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00038     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00039     0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
```

```

00040 0xffff, 0xffff, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00041 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00042 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00043 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00044 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00045 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00046

```

## 2.35 granasat\_logo.h

```

00001 // Generated by : ImageConverter 565 Online
00002 // Generated from : images.png
00003 // Time generated : Wed, 17 Jul 24 21:08:31 +0200 (Server timezone: CET)
00004 // Image Size : 225x225 pixels
00005 // Memory usage : 101250 bytes
00006
00007
00008 #if defined(__AVR__)
00009 #include <avr/pgmspace.h>
00010 #elif defined(__PIC32MX__)
00011 #define PROGMEM
00012 #elif defined(__arm__)
00013 #define PROGMEM
00014 #endif
00015
00016 const unsigned short granasat_logo[] PROGMEM={0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0x54d8, 0x4497, 0x3476, 0x3456, 0x2c36, 0x2c36, 0x2c36, 0x3456, 0x4497, 0x54d8, 0x6538, 0x6d79,
0x85da, 0xa65b, 0xdf3d, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00017 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xef9e, 0xc6fd, 0xa63b,
0x6d59, 0x44d8, 0x3c77, 0x2c36, 0x2436, 0x2436, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x2c56, 0x3456, 0x3456, 0x2c36, 0x2c36, 0x2436, 0x2436, 0x2c56, 0x4497, 0x4cd8, 0x6d59,
0xa63b, 0xcfld, 0xef9e, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00018 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0x0bb5, 0x2c56, 0x2c56, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2416, 0x1bf5, 0x2c36, 0x2c36,
0x2416, 0x13d5, 0x1bf5, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2416, 0x2416, 0x1bf5,
0x1bf6, 0x1c15, 0x3c97, 0x5d38, 0x9e3b, 0xdf5e, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00019 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0x69c, 0xd73d, 0xa65b, 0x2416, 0x2c36, 0x2436, 0x3c77, 0x5d18, 0x7dba, 0x2c56, 0x2c36, 0x2416,
0x54d8, 0xa67c, 0x7dba, 0x2436, 0x2c36, 0x2c36, 0x2c36, 0x3476, 0x2c36, 0x2416, 0x2c36, 0x3456,
0x3456, 0x3456, 0x2c36, 0x2416, 0x1bf5, 0x2416, 0x54f8, 0x961a, 0xdf5e, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00020 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,

```

```
00021 0xf7df, 0xae7c, 0x5d18, 0x2416, 0x1bf5, 0x2416, 0x2c36, 0x3456, 0x2c56, 0x3456, 0x1bf5, 0xb69c,
0xb69c, 0x13f5, 0x2456, 0x2c36, 0x2c36, 0x1bf6, 0x6538, 0xffdf, 0x7dda, 0x2c36, 0x2416, 0x3c77,
0xdf3d, 0x7579, 0xc6fd, 0x85ba, 0x1bf5, 0x2416, 0x4cd8, 0xcefd, 0x2c36, 0x961a, 0x4cd7, 0x2416,
0x2c36, 0x2c36, 0x1c16, 0x1bf5, 0x2c36, 0x2c36, 0x2416, 0x1bf5, 0x2416, 0x5d18, 0xae7c, 0xf7df,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0x3c97, 0x1bf5, 0x1bf6, 0x2c36, 0x2c56, 0x2c56, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x3477, 0x7579, 0xef9e, 0x6d59, 0x2416, 0x2416, 0x54d8, 0xd71d, 0x13d5, 0x2c36, 0x2416, 0x54f8,
0xdf5e, 0x0354, 0x965b, 0xa65b, 0x1bd5, 0x1bf5, 0x6d79, 0xbedd, 0x13d5, 0xdf3d, 0x4cd7, 0x2416,
0x2416, 0x5d18, 0xae5b, 0x7599, 0x3c97, 0x2416, 0x3456, 0x2c36, 0x2c36, 0x1bf5, 0x1bf5, 0x3c97,
0x8dfa, 0xe77e, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00022 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0x2416, 0x3456, 0x3456, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0xa65b, 0x0bb5, 0x9e3b, 0xbecb, 0x13d5, 0x2c36, 0x3477, 0xdf7e, 0x2c56, 0x2c36, 0x2c36, 0x2416,
0xa65b, 0xc1d, 0xc6fd, 0x44b7, 0x2c36, 0x1bf5, 0x759a, 0xd75e, 0x6579, 0xe77e, 0x2436, 0x2c56,
0x1bd5, 0x9e3b, 0xc6fd, 0x961b, 0xc6fd, 0x2416, 0x2c36, 0x2c36, 0x2c36, 0x3456, 0x3456, 0x2416,
0x1bf5, 0x2c56, 0x7dba, 0xdf5e, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00023 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0x2c56, 0x2c56, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0xdf5e, 0xc1d, 0xc6fd, 0x7d99, 0x1c16, 0x2c36, 0x2c36, 0x44b7, 0x2c36, 0x3456, 0x961b, 0xa65b,
0x2416, 0x44d7, 0x3456, 0x2416, 0x2c36, 0x2c36, 0x3456, 0x3456, 0x961b, 0xa65b, 0x1c16, 0x3436,
0x1c16, 0xdf5e, 0x44b7, 0x3497, 0xdf5e, 0x2416, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x3456, 0x2c36, 0x1bf5, 0x2c56, 0x85da, 0xef9e, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00024 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xa67b, 0x3477, 0x1bf5, 0x2c36, 0x3456, 0x2416, 0x2436,
0x2416, 0x2416, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2416,
0x2c56, 0x4cf8, 0x3477, 0x1bf5, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2415, 0x2c36, 0x2c36, 0x2c36,
0x2c36, 0x2415, 0x2c16, 0x2c36, 0x2c36, 0x2c36, 0x2c16, 0x13f5, 0x1bf5, 0x2416, 0x2c36, 0x2416,
0x44b7, 0xe75e, 0xbecb, 0xe79e, 0x7d9a, 0x1bf5, 0x3456, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x2c36, 0x2c36, 0x3456, 0x3456, 0x1bf5, 0x3477, 0xae7c, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00025 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xd73d, 0x54f8, 0x1bf5, 0x2436, 0x3456, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0xbecb, 0x54f8, 0x2416, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x2c16, 0x23f5, 0x2c16, 0x3456, 0x2c57, 0x2c77, 0x2c77, 0x3498, 0x3498, 0x3498, 0x2c98, 0x2c78,
0x2c98, 0x3498, 0x3498, 0x2c98, 0x2c78, 0x2c77, 0x2c77, 0x3477, 0x3456, 0x2c36, 0x3436, 0x13d5,
0x8dda, 0xae7c, 0x3477, 0x5518, 0x2416, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x54f8, 0xd73d, 0xffff, 0xffff, 0xffff,
0x2c36, 0x2c36, 0x2c36, 0x1bf5, 0x2416, 0x13d5, 0x0bb5, 0x54f8, 0xd73d, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00026 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x2436, 0x1bf6, 0x3456, 0x2c36, 0x2c16, 0x3456,
0xc6dd, 0x7599, 0x1bf5, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c57,
0x3477, 0x3498, 0x2c77, 0x2c15, 0x2393, 0x2351, 0x1acf, 0x1a4c, 0x11ea, 0x1188, 0x0947, 0x0947,
0x0947, 0x0947, 0x1188, 0x120a, 0x1a6d, 0x1acf, 0x2351, 0x2393, 0x2c15, 0x2c77, 0x3498, 0x2c57,
0x44b8, 0x44b7, 0x1bf5, 0x2415, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x2c36, 0x2c36, 0x2416, 0x7dba, 0xc6fd, 0xc6dc, 0x5d38, 0x13d5, 0x2436, 0x95fa, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00027 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xdf5e, 0x4cf8, 0x13d5, 0x2c36, 0x3456, 0x2416, 0x3456, 0x3477, 0x2c36, 0x1bf5, 0x8dda, 0xf7bf,
0x5d38, 0x8dfa, 0x2c56, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c57, 0x2c56, 0x2c36, 0x23b3,
0x1aef, 0x11a9, 0x0926, 0x0083, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x2393, 0x2436, 0x3498, 0x2c77, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x3436, 0x1bf5, 0x7d99, 0xefbe, 0x4cd8, 0x5518, 0xc1d, 0x4cd7, 0x2416, 0x1bd5, 0x4cf8, 0xdf5e,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00028 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0x2c36, 0x1c16, 0x3456, 0x2c36, 0x2416, 0x44b7, 0xc1d, 0x5d18, 0x2416, 0x2c36, 0x2436, 0x2c36,
0xb6bc, 0x6d59, 0x2416, 0x2c36, 0x2c36, 0x2c57, 0x3498, 0x2c36, 0x2372, 0x120a, 0x0905, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0926, 0x122b, 0x2372, 0x2c36, 0x3498, 0x2c57, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x2c36, 0x1bf5, 0xdf3e, 0x5518, 0x4cd8, 0x4cf8, 0x2c56, 0x3476, 0x2c36, 0x3456, 0x1c16, 0x2c36,
0xb6bc, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
```

Generated by Doxygen

Generated by Doxygen



Generated by Doxygen

Generated by Doxygen

```
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x23b3, 0x2c57, 0x2c36, 0x2416, 0x1bf5, 0x0395, 0x5d18, 0xae7b, 0x2416, 0x1bf6, 0xa65b, 0xffff,
00064 0xffff, 0x6d59, 0x2416, 0x1bf6, 0x7599, 0xefbf, 0xae7b, 0xdf3d, 0xd73e, 0x5d18, 0x2457, 0x22ef,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x3b0e, 0x44f9,
0x0926, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x2124, 0x8430, 0x39e7, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x18e3, 0x2945, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x2331, 0x1a6d, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x1aef, 0x2c77, 0x2c36, 0x54f8, 0x75ba, 0x8dfa, 0xefd, 0xcf1d, 0x2436, 0x1bf5, 0x6d59, 0xffff,
00065 0xffdf, 0x4cd8, 0x2416, 0x2416, 0x5518, 0xdf3e, 0x9e3b, 0xae7b, 0x3cb7, 0x0b94, 0x3498, 0x11c9,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0020, 0x0000, 0x00a3, 0x2c78,
0x2331, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x18c3, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0841, 0x5aeb, 0x738e, 0x1082,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x1a6d, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x11c9, 0x2c78, 0x3456, 0xc6bc, 0xc6dd, 0xa65b, 0x7d99, 0xdf3d, 0x3c97, 0x1bf5, 0x44d8, 0xffff,
00066 0xdf3d, 0x4497, 0x2416, 0x3456, 0x2416, 0x2416, 0x8dfa, 0xefbf, 0xa67b, 0x2c36, 0x2c57, 0x0926,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x2310,
0x2c78, 0x00c4, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1082, 0x18e3, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0926, 0x2c57, 0x2c36, 0x2c36, 0x13d5, 0x13d5, 0x1bf5, 0x7579, 0x3c97, 0x2416, 0x3c77, 0xf7df,
00067 0xa65b, 0x2c56, 0x2c36, 0x2c36, 0x2c56, 0x2c36, 0x13d5, 0x3477, 0xae5b, 0x4cd8, 0x23f5, 0x00a3,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0926,
0x34b9, 0x1a8d, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0083, 0x2c15, 0x2c36, 0x2c36, 0x3456, 0x3456, 0x3456, 0x2416, 0x2c36, 0x2c36, 0x2416, 0xdf7e,
00068 0x85da, 0x2436, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x3456, 0x2416, 0x1bd5, 0x2c57, 0x23b3, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x2351, 0x2c57, 0x0062, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0041, 0x0020, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0021, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0041, 0x2c15, 0x120b,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0020, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0020, 0x0020, 0x0020, 0x0021, 0x0021, 0x0020, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x2393, 0x2c57, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x3456, 0x2c36, 0x3456, 0x13d5, 0xc6fd,
00069 0x6d79, 0x2436, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c77, 0x2351, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0946, 0x1aae, 0x2331, 0x23b3, 0x23b3,
0x2372, 0x2393, 0x1a8d, 0x0946, 0x2351, 0x2330, 0x2372, 0x2372, 0x2351, 0x1af0, 0x1aae, 0x11a8,
0x08c4, 0x34b9, 0x1a8d, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x124c, 0x23b3, 0x2372, 0x08e5, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x11a9,
0x23b3, 0x0083, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1a8d, 0x2393, 0x2372, 0x00a3, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x11ea, 0x2351, 0x23b4, 0x2bd4, 0x2bd4, 0x2393, 0x2372, 0x2c77,
0x1188, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1aef, 0x23b3, 0x2331, 0x0021, 0x0000, 0x0000, 0x0000,
0x0947, 0x2c16, 0x2bf5, 0x2bf5, 0x23b3, 0x2393, 0x23b4, 0x2bf5, 0x2bf5, 0x2bd4, 0x00a3, 0x0000,
0x0000, 0x2351, 0x2c77, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x1bf5, 0xa65b,
00070 0x6538, 0x2436, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c77, 0x1acf, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x11c9, 0x2c77, 0x3498, 0x3498, 0x2372, 0x1a6d,
0x1a8d, 0x2bd4, 0x2372, 0x11ea, 0x34b9, 0x2c77, 0x2351, 0x2330, 0x23b3, 0x2c98, 0x3498,
0x122b, 0x1aae, 0x3498, 0x0083, 0x0000, 0x0000, 0x0041, 0x2bf5, 0x2c77, 0x34b8, 0x1a6c, 0x0000,
0x0000, 0x0000, 0x0000, 0x1a8d, 0x2c98, 0x3498, 0x2330, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x122b,
0x3498, 0x08a4, 0x0000, 0x0000, 0x0000, 0x0062, 0x2c36, 0x2c77, 0x34b8, 0x11ea, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x1a8d, 0x34b9, 0x2c16, 0x2310, 0x1a4c, 0x1a6d, 0x2393, 0x23b3, 0x11ea,
0x2c78, 0x0905, 0x0000, 0x0000, 0x08e4, 0x2c78, 0x2c57, 0x3498, 0x0947, 0x0000, 0x0000, 0x0000,
0x00a3, 0x1a4c, 0x1a4c, 0x122b, 0x2bd4, 0x3498, 0x2372, 0x120b, 0x1a6c, 0x122c, 0x0041, 0x0000,
0x0000, 0x1acf, 0x2c77, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x1bf6, 0x95fa,
00071 0x54d8, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c78, 0x1a4c, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x08e5, 0x2c57, 0x2bf5, 0x0926, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0062, 0x11c9, 0x2c57, 0x2c36, 0x00a3, 0x0000, 0x0000, 0x08c4, 0x23b3, 0x2c57,
0x2c36, 0x1188, 0x2c56, 0x1aef, 0x0000, 0x0000, 0x0947, 0x2c77, 0x2bf5, 0x2c57, 0x23b3, 0x0000,
0x0000, 0x0000, 0x0000, 0x1a6d, 0x2c77, 0x2c15, 0x2c98, 0x0968, 0x0000, 0x0000, 0x0000, 0x120b,
0x2c77, 0x00a3, 0x0000, 0x0000, 0x0000, 0x11a8, 0x2c77, 0x2bf4, 0x2c57, 0x2372, 0x0000, 0x0000,
0x0000, 0x0000, 0x00a3, 0x2c36, 0x2c36, 0x00c4, 0x0000, 0x0000, 0x0000, 0x08c4, 0x0000,
0x122b, 0x2c36, 0x0082, 0x0000, 0x1a4c, 0x2c57, 0x2bf5, 0x2c77, 0x1acf, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x2331, 0x34b9, 0x1a8d, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x1a4c, 0x2c78, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2416, 0x7559,
00072 0x4497, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x3498, 0x11ea, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1aef, 0x2c77, 0x2c57, 0x0967, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x11a9, 0x2c77, 0x2c36, 0x08c4, 0x0000, 0x0000, 0x0000, 0x0947, 0x2c57,
0x3498, 0x0967, 0x11a9, 0x34d9, 0x0905, 0x0000, 0x1b10, 0x23b3, 0x122b, 0x2c77, 0x2c57, 0x0926,
0x0000, 0x0000, 0x0000, 0x1a6d, 0x2c78, 0x2c16, 0x2c57, 0x2bd4, 0x0041, 0x0000, 0x0000, 0x122b,
```

```

0x2c77, 0x08a4, 0x0000, 0x0000, 0x0000, 0x2351, 0x2392, 0x1a4c, 0x2c77, 0x2c36, 0x08c4, 0x0000,
0x0000, 0x0000, 0x0947, 0x2c77, 0x2c16, 0x0083, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x2372, 0x23b4, 0x0041, 0x2bd4, 0x1aef, 0x1aae, 0x3498, 0x2bd4, 0x0062, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b8, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x11ea, 0x3498, 0x2c16, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
00073 0x3456, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x0000, 0x0000, 0x0000, 0x0000, 0x0082, 0x2bf4, 0x2c77, 0x2372, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x11c9, 0x2c77, 0x2c36, 0x08c4, 0x0000, 0x0020, 0x0000, 0x0083, 0x2c36,
0x3498, 0x11c9, 0x0000, 0x23b3, 0x23d4, 0x0041, 0x2c15, 0x1a8e, 0x0041, 0x2c36, 0x2c98, 0x1a6d,
0x0000, 0x0000, 0x0000, 0x1a6d, 0x2c77, 0x2c36, 0x2c36, 0x2c98, 0x1a8d, 0x0000, 0x0000, 0x120b,
0x2c77, 0x08a4, 0x0000, 0x0000, 0x0041, 0x2c77, 0x122b, 0x0083, 0x2c57, 0x2c78, 0x122b, 0x0000,
0x0000, 0x0000, 0x0905, 0x2c57, 0x2c57, 0x2372, 0x0041, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0062, 0x2c15, 0x2351, 0x2c77, 0x0967, 0x0946, 0x2c78, 0x2c77, 0x0967, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b9, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0988, 0x2c78, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x4497,
00074 0x3456, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c16, 0x2c98, 0x0947, 0x0000,
0x0000, 0x0000, 0x0000, 0x0905, 0x2c56, 0x2c78, 0x122b, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x11c9, 0x2c77, 0x2c36, 0x08c4, 0x0000, 0x0000, 0x0000, 0x120a, 0x2c57,
0x2c77, 0x0967, 0x0000, 0x08e4, 0x2c57, 0x2352, 0x2c77, 0x0926, 0x0000, 0x2331, 0x2c98, 0x23d4,
0x0020, 0x0000, 0x0000, 0x1a8d, 0x2bf4, 0x1acf, 0x2c77, 0x2c36, 0x2c57, 0x08e4, 0x0000, 0x122b,
0x2c77, 0x08a3, 0x0000, 0x0000, 0x11a8, 0x34b8, 0x08e4, 0x0000, 0x2372, 0x2c98, 0x2393, 0x0000,
0x0000, 0x0000, 0x0000, 0x1b10, 0x3498, 0x2c57, 0x23b3, 0x08e5, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0946, 0x2c78, 0x2bf5, 0x0041, 0x0041, 0x2bf5, 0x2c78, 0x1aef, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b9, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0947, 0x2c98, 0x2c16, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x3457,
00075 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c78, 0x0947, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0926, 0x0967, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x11a9, 0x2c77, 0x2c36, 0x0926, 0x00a3, 0x0967, 0x2310, 0x2c78, 0x34b8,
0x2372, 0x0020, 0x0000, 0x0000, 0x11a9, 0x3498, 0x2393, 0x0062, 0x0000, 0x11ea, 0x2c77, 0x2c57,
0x0926, 0x0000, 0x0000, 0x0000, 0x1a8d, 0x2c15, 0x00a3, 0x2bf4, 0x2c56, 0x2c77, 0x0000, 0x122b,
0x2c77, 0x08a4, 0x0000, 0x0000, 0x2310, 0x2bd4, 0x0020, 0x0000, 0x122c, 0x2c78, 0x3436, 0x0105,
0x1000, 0x4000, 0x0800, 0x0021, 0x1af0, 0x2c78, 0x2c57, 0x2c36, 0x11ea, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0020, 0x23b3, 0x2c36, 0x00a3, 0x0000, 0x1aef, 0x2c78, 0x0062, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b9, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000,
00076 0x0000, 0x0947, 0x2c78, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c98, 0x0947, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0926, 0x2c57, 0x2c77, 0x0967, 0x0000, 0x0000, 0x0000, 0x0000,
0x0062, 0x11c9, 0x0926, 0x11c9, 0x2c78, 0x2bf5, 0x2393, 0x2c36, 0x2c98, 0x2c36, 0x2331, 0x120b,
0x0041, 0x0000, 0x0000, 0x0000, 0x0021, 0x2bf5, 0x23d4, 0x0020, 0x0000, 0x0082, 0x2c15, 0x2c98,
0x1a8d, 0x0000, 0x0000, 0x1a8d, 0x2c36, 0x0000, 0x11a8, 0x2c98, 0x2c15, 0x3498, 0x1188, 0x11ea,
0x2c78, 0x00c4, 0x0000, 0x0041, 0x2c57, 0x1a6d, 0x0000, 0x0000, 0x08c4, 0x3436, 0x1cb9, 0x3a4c,
0x8000, 0x1800, 0x0000, 0x0000, 0x0000, 0x1a4c, 0x2c57, 0x2c98, 0x1af0, 0x0021, 0x0000,
0x0000, 0x0000, 0x08e4, 0x2c36, 0x2c77, 0x2372, 0x0000, 0x0967, 0x2c77, 0x2c77, 0x11a9, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b9, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00077 0x0000, 0x0947, 0x2c98, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x3456, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c98, 0x0947, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0905, 0x2c36, 0x2c98, 0x1188, 0x0000, 0x0000, 0x0000,
0x0926, 0x34d9, 0x2393, 0x11ea, 0x2c78, 0x2bf4, 0x11c9, 0x2c36, 0x2c36, 0x23b3, 0x0041, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0947, 0x2c57, 0x2c57, 0x1aae, 0x0000, 0x0000, 0x2310, 0x3498,
0x2bd4, 0x0041, 0x0000, 0x1a8d, 0x2c36, 0x0041, 0x0000, 0x2372, 0x2c77, 0x2c56, 0x23b3, 0x1a8d,
0x2c57, 0x08c4, 0x0000, 0x11a9, 0x3498, 0x0905, 0x0000, 0x0000, 0x0020, 0x13b3, 0x6372, 0xa968,
0x1821, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0946, 0x2bf4, 0x2c57, 0x2c98, 0x1aef, 0x0000,
0x0000, 0x0000, 0x1a6d, 0x2c16, 0x11c9, 0x34b8, 0x11ca, 0x0020, 0x2bf5, 0x2c98, 0x2310, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b9, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00078 0x0000, 0x0947, 0x2c98, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x3457,
0x3456, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x0000, 0x0000, 0x0000, 0x0000, 0x0083, 0x2bf5, 0x3498, 0x1acf, 0x0000, 0x0000, 0x0000, 0x0000,
0x0926, 0x3498, 0x2351, 0x11e9, 0x2c77, 0x2c36, 0x0062, 0x11a9, 0x2c77, 0x2c77, 0x2330, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x1aef, 0x2bf5, 0x0946, 0x2c78, 0x11ea, 0x0000, 0x11ea, 0x2c77,
0x2c57, 0x0947, 0x0000, 0x1a8d, 0x2c36, 0x0062, 0x0000, 0x0905, 0x2c77, 0x2c36, 0x2c36, 0x2c36,
0x2c16, 0x08c4, 0x0000, 0x2330, 0x2bf5, 0x0000, 0x0000, 0x0000, 0x0000, 0x80e4, 0xd126, 0x3bf5,
0x0126, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0082, 0x2351, 0x2c57, 0x2c77, 0x08e5,
0x0000, 0x0021, 0x23d4, 0x2352, 0x0000, 0x1aef, 0x3498, 0x0062, 0x1a8d, 0x2c15, 0x00a3,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b9, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00079 0x0000, 0x1188, 0x3478, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x4497, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c16, 0x3498, 0x11ea, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x2331, 0x2c78, 0x2bf5, 0x0062, 0x0000, 0x0020, 0x0000,
0x0946, 0x34b9, 0x2351, 0x11c9, 0x2c77, 0x2c36, 0x08c4, 0x0000, 0x1a8d, 0x2c77, 0x2c77, 0x1a4c,
0x0000, 0x0000, 0x0000, 0x0021, 0x2c36, 0x1acf, 0x0000, 0x1188, 0x2c78, 0x08c4, 0x0041, 0x2c36,
0x2c98, 0x1ace, 0x0000, 0x1a8d, 0x2c36, 0x0041, 0x0000, 0x0000, 0x1a8d, 0x2c98, 0x2c15, 0x2c56,
0x2c36, 0x00a3, 0x0000, 0x2c77, 0x1a6d, 0x0000, 0x0000, 0x1800, 0xb000, 0xe820, 0x4bb3, 0x24b9,
0x1a6c, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0062, 0x2c36, 0x2c77, 0x11c9,
0x0000, 0x08e4, 0x34b9, 0x11c9, 0x0000, 0x0083, 0x2c78, 0x1aef, 0x0926, 0x2c77, 0x2c77, 0x11a9,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b9, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00080 0x0000, 0x120a, 0x3498, 0x2c16, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x54d8, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c78, 0x1a4c, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0967, 0x0000, 0x0000, 0x2c57, 0x1af0, 0x0020, 0x0000, 0x0000,
0x0905, 0x34b9, 0x2351, 0x11c9, 0x2c77, 0x2c36, 0x08c4, 0x0000, 0x0020, 0x2351, 0x2c77, 0x2c77,
0x1188, 0x0000, 0x0000, 0x0967, 0x34b9, 0x0947, 0x0000, 0x0000, 0x120b, 0x2c36, 0x0041, 0x1aef,
0x3498, 0x2bf4, 0x0000, 0x1a6d, 0x2c36, 0x0041, 0x0000, 0x0000, 0x0062, 0x2c15, 0x2c56, 0x2c36,
0x2c36, 0x0062, 0x0968, 0x34b9, 0x0905, 0x0000, 0x3000, 0xd800, 0xf800, 0x4821, 0x1351, 0x3498,
0x23b3, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0905, 0x2c36, 0x2c57, 0x08e4,
0x0000, 0x1a6d, 0x2c77, 0x0083, 0x0000, 0x0000, 0x1a6c, 0x34b9, 0x0968, 0x2372, 0x3498, 0x2330,
0x0000, 0x0000, 0x0000, 0x0000, 0x2351, 0x34b8, 0x1aae, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x1a6d, 0x2c78, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
00081 0x6538, 0x2436, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1a8d, 0x3498, 0x2c98, 0x2372, 0x11c9, 0x0967,

```

Generated by Doxygen



Generated by Doxygen

	0x0000, 0x0861, 0x39c7, 0x5aeb, 0x4208, 0x0861, 0x0000, 0x0000, 0x0000, 0x39c7, 0x2965, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x73ae, 0xce79, 0x52ab, 0x39e7, 0x1082, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x1a6d, 0x2c98, 0x2c16, 0x3456, 0x2c36, 0x3456, 0x961b, 0xef9e, 0xd73d, 0x2c36,
00108	0x2c36, 0x2416, 0x54f8, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xdf3e, 0x2436, 0x2c36,
	0x2c36, 0x2c36, 0x3456, 0x2c16, 0x2416, 0xae7c, 0x5d18, 0x1bd5, 0x3436, 0x2c98, 0x11ea, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x3186, 0x52aa, 0x4228, 0x18e3, 0x0000, 0x0020, 0x5ac6, 0x39e7,
	0x0020, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1082, 0x2986,
	0x632d, 0x94b3, 0x2966, 0x2124, 0x0841, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x11ea, 0x2c98, 0x2c36, 0x2c36, 0x1bf5, 0x2c36, 0x2c36, 0x13d5, 0x44b7, 0x44b7, 0x2436,
	0x2c36, 0x2436, 0xdf3e, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00109	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x961b, 0x13f5,
	0x3456, 0x2c36, 0x2c36, 0x3456, 0xcf1d, 0xc6fd, 0xa65b, 0x4cd8, 0x2416, 0x2c36, 0x2c77, 0x1188,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0841, 0x5aeb, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x1188, 0x2c77, 0x2c36, 0x2416, 0x3c77, 0x8dfa, 0x2436, 0x2416, 0x3456, 0x2416, 0x2416, 0x3456,
00110	0x13d5, 0x961b, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x54f8,
	0x1bf6, 0x3456, 0x1bd5, 0x85da, 0xcefd, 0x7d99, 0xe77e, 0x85ba, 0x1bf5, 0x3436, 0x2c36, 0x2c56,
	0x0946, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x3186, 0x738e, 0x9492, 0xc638,
	0x4228, 0x2965, 0x10a2, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x1082, 0x39e7, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0946,
00111	0x2c56, 0x2c36, 0x2c36, 0x2c36, 0x2c56, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c36,
	0x4cf8, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xdf7e,
	0x2c56, 0x2c36, 0x2416, 0x4497, 0xb69c, 0x6d59, 0x5d59, 0x2c36, 0x2c36, 0x2c36, 0x2c36, 0x2c56,
	0x2c36, 0x0946, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x10a2, 0x3186,
	0x5aeb, 0x39e7, 0x18e3, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0841, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0946, 0x2c36,
	0x2c56, 0x2c36, 0x2c56, 0x3456, 0x2c36, 0x13d5, 0x95fa, 0xcf1d, 0x3456, 0x3456, 0x2c36, 0x2c36,
00112	0xdf5e, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xb6bc, 0x13f5, 0x3456, 0x2416, 0x4497, 0x3c77, 0x13d5, 0x2c36, 0x2416, 0x3456, 0x3477, 0x2c36,
	0x2c56, 0x2c56, 0x1188, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x738e,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x2c36, 0x2416, 0x1bf6, 0x1bf5, 0x2c36, 0x2c56, 0x1bf5, 0x3456, 0xae7b, 0x5cf8, 0x0bd5, 0xb6bc,
00113	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0x8dfa, 0x13d5, 0x3456, 0x2416, 0x2c36, 0x3456, 0x2416, 0x3c77, 0xe77e, 0x5d18, 0x0bb5,
	0x2c36, 0x2c56, 0x2c57, 0x11ea, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x4a69,
	0x1082, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x11ea, 0x2c57,
	0x23f6, 0x7dba, 0xb6bc, 0x6559, 0x1c16, 0x2c36, 0x3456, 0x2416, 0x3c77, 0x1bf5, 0x8dfa, 0xffff,
00114	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0x6d59, 0x1bf5, 0x3456, 0x2c36, 0x2c36, 0x3477, 0xdf5e, 0x85ba, 0x1bf6, 0x85da,
	0x2c56, 0x2c36, 0x2c36, 0x2c98, 0x1a6d, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0861,
	0x0020, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x1bd5, 0x961b, 0xd73d, 0xb6bc, 0xcefd, 0x4497, 0x2416, 0x3456, 0x13d5, 0x6d59, 0xffff, 0xffff,
00115	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
	0x2416, 0x2c36, 0x2c36, 0x2c98, 0x2c36, 0x2351, 0x0083, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
	0x2c36, 0x3456, 0xcf1d, 0x3497, 0x54f8, 0x3477, 0x2c36, 0x1bf5, 0x4cd8, 0xdf7d, 0xffff, 0xffff,
	0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,



Generated by Doxygen

Generated by Doxygen

[illegible]

## 2.36 images.h

```
00001 // images.h
00002
00003 #ifndef IMAGES_H
00004 #define IMAGES_H
00005
00006 #include "automatic_icon.h"
00007 #include "brightness_icon.h"
00008 #include "color_icon.h"
00009 #include "empty_sun.h"
00010 #include "ethernet_active.h"
00011 #include "ethernet_icon.h"
00012 #include "font_icon.h"
00013 #include "full_sun.h"
00014 #include "granasat_logo.h"
00015 #include "manual_icon.h"
00016 #include "settings_icon.h"
00017 #include "step_icon.h"
00018
00019 #endif // IMAGES_H
```

## 2.37 manual icon.h

```

00001 #if defined(__AVR__)
00002     #include <avr/pgmspace.h>
00003 #elif defined(__PIC32MX__)
00004     #define PROGMEM
00005 #elif defined(__arm__)
00006     #define PROGMEM
00007 #endif
00008
00009 const unsigned short manual_icon[] PROGMEM={0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00010 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00011 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00012 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00013 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00014 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,

```

```

00015 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
00016 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x2104, 0xbd7f, 0x2945, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00017 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x10a2, 0x0000, 0x0000, 0x0020, 0x0000, 0x630c, 0xffff, 0xbd7f, 0xffff, 0x6b6d, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00018 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x1082,
      0xa514, 0xf79e, 0x5acb, 0x0000, 0x0000, 0x10a2, 0xef5d, 0x8430, 0x0000, 0x7bcf, 0xf79e, 0x18c3,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00019 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x39e7, 0xef7d,
      0xffff, 0xffff, 0xffff, 0xbd7f, 0x0841, 0x1082, 0xdefb, 0xa534, 0x0020, 0x9cd3, 0xe73c, 0x10a2,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
00020 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x2104, 0xf79e, 0xffff,
      0x6b6d, 0x5aeb, 0x9cd3, 0xffff, 0xbd7f, 0x0000, 0x4a49, 0xf79e, 0xe71c, 0xf79e, 0x52aa, 0x0000,
      0x0020, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00021 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x9cd3, 0xffff, 0x4a69,
      0x0000, 0x0000, 0x0000, 0xa534, 0xffff, 0x4208, 0x0000, 0x0841, 0x8c51, 0x0861, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00022 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xd69a, 0xffff, 0x18c3,
      0x0020, 0x0861, 0x0000, 0x738e, 0xffff, 0x6b6d, 0x0000, 0x0000, 0x31a6, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00023 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x6b4d, 0xffff, 0x39e7,
      0x0000, 0x0000, 0x0000, 0xa514, 0xffff, 0x18c3, 0x0861, 0x94b2, 0xd69a, 0x9cf3, 0x0861, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00024 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0841, 0xbd7f, 0xffff,
      0x5acb, 0x4228, 0x8c71, 0xffff, 0x738e, 0x0000, 0x39e7, 0xe73c, 0x0000, 0xdedb, 0x4a49, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00025 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x94b2,
      0xffff, 0xffff, 0xe71c, 0x630c, 0x0000, 0x0000, 0x0000, 0x08c71, 0xd69a, 0x94b2, 0x0861, 0x0020,
      0x0020, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00026 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x6b4d, 0xbd7f,
      0xef7d, 0xffff, 0xdefb, 0xc618, 0xef5d, 0xd6ba, 0x2104, 0x0000, 0x39e7, 0x0020, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
00027 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x8c71, 0xdedb, 0xd6ba, 0xdedb, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0x9cd3, 0x9cd3, 0x4208, 0x0020, 0x0000, 0x0000, 0x0000, 0x4a49,
      0x6b6d, 0x0861, 0x0000, 0x0000, 0x0000,
00028 0xffff, 0x0000, 0x0000, 0x0020, 0x0000, 0x0000, 0xad55, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xef7d, 0x52aa, 0x4208, 0x0000, 0x0000, 0x0000, 0x4a49, 0xce79, 0xffff,
      0xbdd7, 0x1082, 0x0000, 0x0000, 0xffff,
00029 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xa534, 0xffff, 0xffdf, 0xffdf, 0xffdf, 0xffdf, 0xffdf,
      0xffdf, 0xffdf, 0xffdf, 0xffdf, 0xffdf, 0x9cd3, 0x6b4d, 0xce59, 0xffff, 0xe71c, 0x632c,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
00030 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0x0020, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00031 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0xa514, 0xffff, 0xffff, 0xd6ba, 0x7bef, 0x7bef,
      0x7bef, 0x7bef, 0x7bef, 0x7bef, 0x7bef, 0x73ae, 0x7bcf, 0x2104, 0x0000, 0x0000, 0x0020,
      0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
00032 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x18c3, 0x73ae, 0x73ae, 0x18c3, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
00033 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0020, 0x0020,
      0x0020, 0x0020, 0x0020, 0x0020, 0x0020, 0x0020, 0x0020, 0x0020, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
00034 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0020, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
00035 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
00036 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00037 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff,
00038 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff};

```

## 2.38 settings\_icon.h

```

00001 #if defined(__AVR__)
00002     #include <avr/pgmspace.h>
00003 #elif defined(__PIC32MX__)
00004     #define PROGMEM
00005 #elif defined(__arm__)
00006     #define PROGMEM
00007 #endif
00008

```

Generated by Doxygen



```

00032 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
00033 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff,
00034 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00035 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00036 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
00037 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0x0000, 0x0000, 0x0000,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff,
00038 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff,
      0xffff, 0xffff, 0xffff, 0xffff, 0xffff, 0xffff);

```

## 2.40 libraries.h File Reference

This header file groups all the libraries and external dependencies required for the project. Including this file in the code ensures the correct inclusion of all necessary libraries, simplifying dependency management and keeping the main code clean and organized.

```

#include <Arduino.h>
#include <freertos/FreeRTOS.h>
#include <freertos/task.h>
#include <BfButton.h>
#include <TFT_eSPI.h>
#include <EthernetENC.h>
#include <SPI.h>
#include <ThreeWire.h>
#include <RtcDS1302.h>
#include <AccelStepper.h>
#include "icons/images.h"
#include <custom_fonts/custom_fonts.h>

```

### 2.40.1 Detailed Description

This header file groups all the libraries and external dependencies required for the project. Including this file in the code ensures the correct inclusion of all necessary libraries, simplifying dependency management and keeping the main code clean and organized.

#### Author

Juan Alberto Serrano Redondo

#### Included libraries:

- **Arduino.h:** Provides essential functions for interacting with Arduino hardware.
- **FreeRTOS:** Enables the use of the real-time operating system (RTOS) on compatible microcontrollers.
- **BfButton:** Provides advanced functionalities for handling buttons.

- TFT\_eSPI: Facilitates communication and control of TFT displays.
- EthernetENC: Provides support for Ethernet communication using the ENC28J60 chip.
- SPI: Serial Peripheral Interface communication protocol.
- ThreeWire: Library for communication on three-wire buses.
- RtcDS1302: Provides functions for interacting with the DS1302 RTC module.
- AccelStepper: Enables control of stepper motors.
- Icons: Defines all icons used in the project's graphical interface.
- Fonts: Defines the fonts used in the project.

Including this file in any part of the project ensures that all necessary dependencies are available and correctly configured.

## 2.41 libraries.h

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

```
00001
00025 #ifndef LIBRARIES_H
00026 #define LIBRARIES_H
00027
00028 #include <Arduino.h>
00029 #include <freertos/FreeRTOS.h>
00030 #include <freertos/task.h>
00031 #include <BfButton.h>
00032 #include <TFT_eSPI.h>
00033 #include <EthernetENC.h>
00034 #include <SPI.h>
00035 #include <ThreeWire.h>
00036 #include <RtcDS1302.h>
00037 #include <AccelStepper.h>
00038 #include "icons/images.h" // Includes all necessary icons
00039 #include <custom_fonts/custom_fonts.h>
00040
00041 #endif // LIBRARIES_H
```

## 2.42 motor/motor.cpp File Reference

Implementation file for motor control functions.

```
#include "motor.h"
#include <globals/globals.h>
#include <libraries.h>
```

### Functions

- void [moveMotor](#) (int steps)  
*Moves the stepper motor a specified number of steps.*



### 2.42.1 Detailed Description

Implementation file for motor control functions.

#### Author

Juan Alberto Serrano Redondo

This file contains the implementation of functions used to control the stepper motor, including moving the motor a specified number of steps.

### 2.42.2 Function Documentation

#### 2.42.2.1 moveMotor()

```
void moveMotor (  
    int steps)
```

Moves the stepper motor a specified number of steps.

This function controls the stepper motor by setting the direction pin and generating step pulses to move the motor. It updates the motor's position and refreshes the screen if the current menu is MANUAL or AUTO.

#### Parameters

<i>steps</i>	The number of steps to move the motor. Positive values move the motor in one direction, and negative values move it in the opposite direction.
--------------	--

## 2.43 motor/motor.h File Reference

Header file for motor control functions.

#### Functions

- void `moveMotor` (int steps)  
*Moves the stepper motor a specified number of steps.*

### 2.43.1 Detailed Description

Header file for motor control functions.

#### Author

Juan Alberto Serrano Redondo

This file contains the declarations of functions and variables used for controlling a stepper motor. It defines the necessary pins and functions to move the motor and update the screen based on the motor's position.

## 2.43.2 Function Documentation

### 2.43.2.1 moveMotor()

```
void moveMotor (  
    int steps)
```

Moves the stepper motor a specified number of steps.

This function controls the stepper motor by setting the direction pin and generating step pulses to move the motor. It updates the motor's position and refreshes the screen if the current menu is MANUAL or AUTO.

## Parameters

<i>steps</i>	The number of steps to move the motor. Positive values move the motor in one direction, and negative values move it in the opposite direction.
--------------	--

## 2.44 motor.h

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

```
00001
00011 #ifndef MOTOR_H
00012 #define MOTOR_H
00013
00025 void moveMotor(int steps);
00026
00027 #endif // MOTOR_H
```

## 2.45 project\_tasks/projectTasks.cpp File Reference

Header file for system task management.

```
#include "projectTasks.h"
#include <motor/motor.h>
#include <GUI/GUI.h>
#include <button/button.h>
#include <libraries.h>
#include <globals/globals.h>
```

### Functions

- void [taskShowTime](#) (void \*pvParameters)  
*Task to display the current time on the screen.*
- void [taskStartScreen](#) (void \*pvParameters)  
*Task to initialize and display the startup screen.*
- void [taskUpdateScreen](#) (void \*pvParameters)  
*Task to update the screen based on the current menu.*
- void [taskScreenTimeout](#) (void \*pvParameters)  
*Task to manage screen timeout and automatic brightness adjustment.*
- void [rotarymotorTask](#) (void \*pvParameters)  
*Task for managing the rotary encoder.*
- void [motorTask](#) (void \*pvParameters)  
*Task for controlling the motor.*
- void [taskEthernet](#) (void \*pvParameters)  
*Task for managing Ethernet communication.*
- void [taskEncoder](#) (void \*pvParameters)  
*Task for handling encoder-related tasks.*
- void [taskButtonPress](#) (void \*pvParameters)  
*Task for handling button press events.*

### 2.45.1 Detailed Description

Header file for system task management.

This header file contains function for managing system tasks, user interface on the screen, Ethernet communication, stepper motor control, and rotary encoder handling.

#### Author

Juan Alberto Serrano Redondo.

### 2.45.2 Function Documentation

#### 2.45.2.1 motorTask()

```
void motorTask (
    void * pvParameters)
```

Task for controlling the motor.

This task adjusts the motor speed based on the rotary encoder's state. It operates only when the menu is set to AUTO.

#### Parameters

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

#### 2.45.2.2 rotarymotorTask()

```
void rotarymotorTask (
    void * pvParameters)
```

Task for managing the rotary encoder.

This task reads the state of the rotary encoder and updates the position and display accordingly. It only operates when the menu is set to AUTO.

#### Parameters

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

#### 2.45.2.3 taskButtonPress()

```
void taskButtonPress (
    void * pvParameters)
```

Task for handling button press events.

This task processes button presses, including single, double, and long presses, and executes the corresponding handlers.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.45.2.4 taskEncoder()**

```
void taskEncoder (  
    void * pvParameters)
```

Task for handling encoder-related tasks.

This task processes the rotary encoder input to navigate menus and adjust settings. It updates the display based on the current menu and encoder position.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.45.2.5 taskEthernet()**

```
void taskEthernet (  
    void * pvParameters)
```

Task for managing Ethernet communication.

This task handles Ethernet client connections and commands. It provides a menu for controlling the motor and updates the Ethernet status.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.45.2.6 taskScreenTimeout()**

```
void taskScreenTimeout (  
    void * pvParameters)
```

Task to manage screen timeout and automatic brightness adjustment.

Handles screen timeout and controls brightness and automatic shutdown.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.45.2.7 taskShowTime()**

```
void taskShowTime (  
    void * pvParameters)
```

Task to display the current time on the screen.

This task shows the current time on the screen if the current menu is not the initial screen.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.45.2.8 taskStartScreen()**

```
void taskStartScreen (  
    void * pvParameters)
```

Task to initialize and display the startup screen.

Configures and shows the initial screen, then switches to the main menu.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.45.2.9 taskUpdateScreen()**

```
void taskUpdateScreen (  
    void * pvParameters)
```

Task to update the screen based on the current menu.

Updates the screen according to the current menu and commands received in the queue.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.46 project\_tasks/projectTasks.h File Reference**

Header file for system task management.

```
#include <libraries.h>
```

## Functions

- void [taskUpdateScreen](#) (void \*pvParameters)  
*Task to update the screen based on the current menu.*
- void [taskStartScreen](#) (void \*pvParameters)  
*Task to initialize and display the startup screen.*
- void [taskShowTime](#) (void \*pvParameters)  
*Task to display the current time on the screen.*
- void [taskScreenTimeout](#) (void \*pvParameters)  
*Task to manage screen timeout and automatic brightness adjustment.*
- void [rotarymotorTask](#) (void \*pvParameters)  
*Task for managing the rotary encoder.*
- void [motorTask](#) (void \*pvParameters)  
*Task for controlling the motor.*
- void [taskEthernet](#) (void \*pvParameters)  
*Task for managing Ethernet communication.*
- void [taskEncoder](#) (void \*pvParameters)  
*Task for handling encoder-related tasks.*
- void [taskButtonPress](#) (void \*pvParameters)  
*Task for handling button press events.*

### 2.46.1 Detailed Description

Header file for system task management.

This header file contains function declarations for managing system tasks, user interface on the screen, Ethernet communication, stepper motor control, and rotary encoder handling.

#### Note

Ensure necessary libraries and definitions are included in the corresponding .cpp file.

Adjust constants and configurations according to hardware requirements.

### 2.46.2 Function Documentation

#### 2.46.2.1 motorTask()

```
void motorTask (
    void * pvParameters)
```

Task for controlling the motor.

This task adjusts the motor speed based on the rotary encoder's state. It operates only when the menu is set to AUTO.

#### Parameters

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

#### 2.46.2.2 rotarymotorTask()

```
void rotarymotorTask (
    void * pvParameters)
```

Task for managing the rotary encoder.

This task reads the state of the rotary encoder and updates the position and display accordingly. It only operates when the menu is set to AUTO.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.46.2.3 taskButtonPress()**

```
void taskButtonPress (  
    void * pvParameters)
```

Task for handling button press events.

This task processes button presses, including single, double, and long presses, and executes the corresponding handlers.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.46.2.4 taskEncoder()**

```
void taskEncoder (  
    void * pvParameters)
```

Task for handling encoder-related tasks.

This task processes the rotary encoder input to navigate menus and adjust settings. It updates the display based on the current menu and encoder position.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.46.2.5 taskEthernet()**

```
void taskEthernet (  
    void * pvParameters)
```

Task for managing Ethernet communication.

This task handles Ethernet client connections and commands. It provides a menu for controlling the motor and updates the Ethernet status.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.46.2.6 taskScreenTimeout()**

```
void taskScreenTimeout (  
    void * pvParameters)
```

Task to manage screen timeout and automatic brightness adjustment.

Handles screen timeout and controls brightness and automatic shutdown.



**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.46.2.7 taskShowTime()**

```
void taskShowTime (  
    void * pvParameters)
```

Task to display the current time on the screen.

This task shows the current time on the screen if the current menu is not the initial screen.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.46.2.8 taskStartScreen()**

```
void taskStartScreen (  
    void * pvParameters)
```

Task to initialize and display the startup screen.

Configures and shows the initial screen, then switches to the main menu.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

**2.46.2.9 taskUpdateScreen()**

```
void taskUpdateScreen (  
    void * pvParameters)
```

Task to update the screen based on the current menu.

Updates the screen according to the current menu and commands received in the queue.

**Parameters**

<i>pvParameters</i>	Pointer to task parameters (not used).
---------------------	--

## 2.47 projectTasks.h

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

```
00001
00013 #ifndef PROJECTTASK_H
00014 #define PROJECTTASK_H
00015 #include <libraries.h>
00016
00017 // LCD tasks
00018 void taskUpdateScreen(void *pvParameters);
00019 void taskStartScreen(void *pvParameters);
00020 void taskShowTime(void *pvParameters);
00021 void taskScreenTimeout(void *pvParameters);
00022
00023 // Motor tasks
00024 void rotarymotorTask(void *pvParameters);
00025 void motorTask(void *pvParameters);
00026
00027 // Ethernet tasks
00028 void taskEthernet(void *pvParameters);
00029
00030 // Rotary encoder tasks
00031 void taskEncoder(void *pvParameters);
00032 void taskButtonPress(void *pvParameters);
00033
00034 #endif // PROJECTTASK_H
```

## 2.48 setup/setup.h File Reference

Contains all setup configurations for the project including Ethernet, LCD, Motor, Pin, RTC, and Task setups.

```
#include <globals/globals.h>
#include <libraries.h>
#include <GUI/GUI.h>
#include <project_tasks/projectTasks.h>
```

### Functions

- void [setup](#) ()

### 2.48.1 Detailed Description

Contains all setup configurations for the project including Ethernet, LCD, Motor, Pin, RTC, and Task setups.

### 2.48.2 Function Documentation

#### 2.48.2.1 setup()

```
void setup ()
```

Ethernet setup configuration for ENC28J60 module.

Initializes SPI communication and configures the Ethernet with the provided MAC address and IP.

< Initializes SPI communication

< CS pin for ENC28J60 module

< Start Ethernet with MAC address and IP

LCD setup configuration for the TFT display.

Configures serial communication, initializes the display, sets text properties, and creates necessary queues and semaphores for screen management.

< Initialize serial communication at 115200 baud

< Initialize TFT display

< Disable text wrapping

< Set text size to small

< Set display rotation

< Create a queue for screen updates

< Set initial screen brightness

< Enable byte swapping for better display rendering

< Create a mutex for controlling screen updates

Motor setup configuration.

Configures motor speed, acceleration, and reads initial state from encoder.

< Read initial state of the encoder

< Set maximum motor speed

< Set motor acceleration

Pin setup configuration.

Configures the pins for the encoder and motor control.

< Set CLK pin as input with pull-up resistor

< Set DT pin as input with pull-up resistor

< Set direction pin as output

< Set step pin as output

RTC setup configuration.

Initializes the Real-Time Clock (RTC) and sets the date and time based on the compilation time. Verifies if the RTC is write-protected and ensures it is running.

< Initialize RTC communication

< Display the compilation date

< Display the compilation time

< Set compiled date and time

Check if the RTC's date and time are valid, and if not, set the compiled date and time.

< Set RTC to compilation time

Check if the RTC is write-protected, and disable write protection if necessary.

< Disable write protection

Check if the RTC is running, and start it if it is not.

< Start RTC

< Get the current date and time from the RTC

Compare the RTC time with the compiled time and update the RTC if necessary.

< Update RTC to compiled date and time

Task setup configuration.

Creates various FreeRTOS tasks for managing display, button presses, Ethernet, motor control, and screen updates.

< Create task for displaying time

< Create task for starting the screen (core 0)

< Create task for reading encoder (core 0)

< Create task for managing screen timeout

< Create task for handling button presses

< Create task for Ethernet communication

< Create task for updating screen (core 1)

< Create task for rotary motor control

< Create task for motor control

< Start the FreeRTOS scheduler

## 2.49 setup.h

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

```

00001
00006 #ifndef SETUP_H
00007 #define SETUP_H
00008
00009 #include <globals/globals.h>
00010 #include <libraries.h>
00011 #include <GUI/GUI.h>
00012 #include <project_tasks/projectTasks.h>
00013 void setup(){
00019     SPI.begin();
00020     Ethernet.init(26);
00021     Ethernet.begin(mac, ip);
00029     Serial.begin(115200);
00030     tft.init();
00031     tft.setTextWrap(false);
00032     tft.setTextSize(1);
00033     tft.setRotation(1);
00034     screenUpdateQueue = xQueueCreate(10, sizeof(ScreenUpdateCommand));
00035     setBrightness(brightnessLevel);
00036     tft.setSwapBytes(true);
00037     gatekeeper = xSemaphoreCreateMutex();
00044     aLastState = digitalRead(CLK);
00045     motor.setMaxSpeed(2000.0);
00046     motor.setAcceleration(1000.0);
00053     pinMode(CLK, INPUT_PULLUP);
00054     pinMode(DT, INPUT_PULLUP);
00055     pinMode(DIR_PIN, OUTPUT);
00056     pinMode(STEP_PIN, OUTPUT);
00064     Rtc.Begin();
00066     Serial.print("compiled: ");
00067     Serial.print(__DATE__);
00068     Serial.print(" ");
00069     Serial.println(__TIME__);
00071     RtcDateTime compiled = RtcDateTime(__DATE__, __TIME__);
00076     if (!Rtc.IsDateTimeValid()) {
00077         Serial.println("RTC lost confidence in the DateTime! Setting compilation date and time.");
00078         Rtc.SetDateTime(compiled);
00079     }
00080
00084     if (Rtc.GetIsWriteProtected()) {
00085         Serial.println("RTC was write protected, enabling writing now");
00086         Rtc.SetIsWriteProtected(false);
00087     }
00088
00092     if (!Rtc.GetIsRunning()) {
00093         Serial.println("RTC was not actively running, starting now");
00094         Rtc.SetIsRunning(true);
00095     }
00096
00097     RtcDateTime now = Rtc.GetDateTime();
00102     if (now < compiled) {
00103         Serial.println("RTC is older than compile time! Updating DateTime.");
00104         Rtc.SetDateTime(compiled);
00105     } else if (now > compiled) {
00106         Serial.println("RTC is newer than compile time. (this is expected)");
00107     } else if (now == compiled) {
00108         Serial.println("RTC is the same as compile time! (not expected but all is fine)");
00109     }
00110
00116     xTaskCreatePinnedToCore(taskShowTime, "ShowTime", 16384, NULL, 1, NULL, 1);
00117     xTaskCreatePinnedToCore(taskStartScreen, "TaskScreenStart", 16384, NULL, 5, NULL, 0);
00118     xTaskCreatePinnedToCore(taskEncoder, "TaskEncoder", 16384, NULL, 3, NULL, 0);
00119     xTaskCreatePinnedToCore(taskScreenTimeout, "ScreenTimeout", 16384, NULL, 10, NULL, 1);
00120     xTaskCreatePinnedToCore(taskButtonPress, "ButtonPress", 16384, NULL, 3, NULL, 0);
00121     xTaskCreatePinnedToCore(taskEthernet, "TaskEthernet", 16384, NULL, 7, NULL, 1);
00122     xTaskCreatePinnedToCore(taskUpdateScreen, "UpdateScreen", 16384, NULL, 2, NULL, 0);
00123     xTaskCreate(rotarymotorTask, "Rotary motor Task", 2048, NULL, 4, NULL);
00124     xTaskCreate(motorTask, "Motor Task", 2048, NULL, 6, NULL);
00125     vTaskStartScheduler();
00127     ScreenUpdateCommand initialCommand = UPDATE_MAIN_MENU;
00128     xQueueSend(screenUpdateQueue, &initialCommand, portMAX_DELAY);
00129 }
00130 #endif // COMBINED_SETUP_H

```



# Index

AUTO  
    globals.h, 57

BACKGROUND  
    globals.h, 57

backgroundcolors  
    globals.cpp, 49  
    globals.h, 57

backgroundpossibleColors  
    globals.cpp, 49  
    globals.h, 57

BRIGHTNESS  
    globals.h, 56

button.cpp  
    doublePressHandler, 3  
    longPressHandler, 4  
    singlePressHandler, 4

button.h  
    doublePressHandler, 5  
    longPressHandler, 5  
    singlePressHandler, 6

button/button.cpp, 3  
button/button.h, 4, 6

centerX  
    globals.cpp, 49  
    globals.h, 57

COLOR  
    globals.h, 56

colormenuArray  
    globals.cpp, 49  
    globals.h, 58

currentMenu  
    globals.cpp, 49  
    globals.h, 58

custom\_fonts/arimo12.h, 7  
custom\_fonts/arimo9.h, 9  
custom\_fonts/custom\_fonts.h, 11  
custom\_fonts/dialog12.h, 12  
custom\_fonts/dialog9.h, 14  
custom\_fonts/gothic12.h, 16  
custom\_fonts/gothic9.h, 19  
custom\_fonts/mono12.h, 21  
custom\_fonts/mono9.h, 24  
custom\_fonts/opensans12.h, 26  
custom\_fonts/opensans9.h, 28  
custom\_fonts/roboto12.h, 31  
custom\_fonts/roboto9.h, 33  
custom\_fonts/SansSerif12.h, 35  
custom\_fonts/SansSerif9.h, 38

custom\_fonts/serif12.h, 40  
custom\_fonts/serif9.h, 43

DIR\_PIN  
    globals.cpp, 49  
    globals.h, 58

doublePressHandler  
    button.cpp, 3  
    button.h, 5

drawBrightnessBar  
    GUI.cpp, 63  
    GUI.h, 66

drawDialMarks  
    GUI.cpp, 63  
    GUI.h, 66

drawNeedle  
    GUI.cpp, 63  
    GUI.h, 66

drawSelectionBar  
    GUI.cpp, 64  
    GUI.h, 68

globals.cpp  
    backgroundcolors, 49  
    backgroundpossibleColors, 49  
    centerX, 49  
    colormenuArray, 49  
    currentMenu, 49  
    DIR\_PIN, 49  
    images\_main, 50  
    lastClk, 50  
    LOOP\_TICKS, 50  
    menuIndex, 50  
    minBrightness, 50  
    possibleFont12, 51  
    screenOn, 51  
    screenUpdateQueue, 51  
    selectedBackgroundColor, 51

globals.h  
    AUTO, 57  
    BACKGROUND, 57  
    backgroundcolors, 57  
    backgroundpossibleColors, 57  
    BRIGHTNESS, 56  
    centerX, 57  
    COLOR, 56  
    colormenuArray, 58  
    currentMenu, 58  
    DIR\_PIN, 58  
    images\_main, 58

- INITIAL\_SCREEN, 57
- lastCik, 58
- LOOP\_TICKS, 59
- MAIN\_MENU, 56
- MANUAL, 57
- menuIndex, 59
- MenuState, 56
- minBrightness, 59
- possibleFont12, 59
- screenOn, 59
- ScreenUpdateCommand, 57
- screenUpdateQueue, 60
- selectedBackgroundColor, 60
- SELECTION\_BAR, 57
- SELECTION\_BORDER, 57
- SERVIDOR\_TCP, 57
- SETTINGS, 56
- STEPPER\_MOTOR, 57
- TEXT, 57
- TEXT\_TYPE, 56
- UPDATE\_CONFIG\_ASPECT, 57
- UPDATE\_CONFIG\_BACKGROUND\_COLOR, 57
- UPDATE\_CONFIG\_BRIGHTNESS, 57
- UPDATE\_MAIN\_MENU, 57
- globals/globals.cpp, 45
- globals/globals.h, 52, 60
- GUI.cpp
  - drawBrightnessBar, 63
  - drawDialMarks, 63
  - drawNeedle, 63
  - drawSelectionBar, 64
  - printDateTime, 64
  - setBrightness, 64
  - updateDial, 65
- GUI.h
  - drawBrightnessBar, 66
  - drawDialMarks, 66
  - drawNeedle, 66
  - drawSelectionBar, 68
  - printDateTime, 68
  - setBrightness, 69
  - updateDial, 69
- GUI/GUI.cpp, 62
- GUI/GUI.h, 65, 69
- icons/automatic\_icon.h, 70
- icons/brightness\_icon.h, 71
- icons/color\_icon.h, 72
- icons/empty\_sun.h, 73
- icons/ethernet\_active.h, 74
- icons/ethernet\_icon.h, 75
- icons/font\_icon.h, 76
- icons/full\_sun.h, 78
- icons/granasat\_logo.h, 79
- icons/images.h, 93
- icons/manual\_icon.h, 93
- icons/settings\_icon.h, 94
- icons/step\_icon.h, 96
- images\_main
  - globals.cpp, 50
  - globals.h, 58
- INITIAL\_SCREEN
  - globals.h, 57
- lastCik
  - globals.cpp, 50
  - globals.h, 58
- libraries.h, 97
- longPressHandler
  - button.cpp, 4
  - button.h, 5
- LOOP\_TICKS
  - globals.cpp, 50
  - globals.h, 59
- MAIN\_MENU
  - globals.h, 56
- MANUAL
  - globals.h, 57
- menuIndex
  - globals.cpp, 50
  - globals.h, 59
- MenuState
  - globals.h, 56
- minBrightness
  - globals.cpp, 50
  - globals.h, 59
- motor.cpp
  - moveMotor, 99
- motor.h
  - moveMotor, 100
- motor/motor.cpp, 98
- motor/motor.h, 99, 101
- motorTask
  - projectTasks.cpp, 102
  - projectTasks.h, 105
- moveMotor
  - motor.cpp, 99
  - motor.h, 100
- possibleFont12
  - globals.cpp, 51
  - globals.h, 59
- printDateTime
  - GUI.cpp, 64
  - GUI.h, 68
- project\_tasks/projectTasks.cpp, 101
- project\_tasks/projectTasks.h, 104, 108
- projectTasks.cpp
  - motorTask, 102
  - rotarymotorTask, 102
  - taskButtonPress, 102
  - taskEncoder, 103
  - taskEthernet, 103
  - taskScreenTimeout, 103
  - taskShowTime, 103
  - taskStartScreen, 104
  - taskUpdateScreen, 104



- projectTasks.h
  - motorTask, [105](#)
  - rotarymotorTask, [105](#)
  - taskButtonPress, [106](#)
  - taskEncoder, [106](#)
  - taskEthernet, [106](#)
  - taskScreenTimeout, [106](#)
  - taskShowTime, [107](#)
  - taskStartScreen, [107](#)
  - taskUpdateScreen, [107](#)
- rotarymotorTask
  - projectTasks.cpp, [102](#)
  - projectTasks.h, [105](#)
- screenOn
  - globals.cpp, [51](#)
  - globals.h, [59](#)
- ScreenUpdateCommand
  - globals.h, [57](#)
- screenUpdateQueue
  - globals.cpp, [51](#)
  - globals.h, [60](#)
- selectedBackgroundColor
  - globals.cpp, [51](#)
  - globals.h, [60](#)
- SELECTION\_BAR
  - globals.h, [57](#)
- SELECTION\_BORDER
  - globals.h, [57](#)
- SERVITOR\_TCP
  - globals.h, [57](#)
- setBrightness
  - GUI.cpp, [64](#)
  - GUI.h, [69](#)
- SETTINGS
  - globals.h, [56](#)
- setup
  - setup.h, [108](#)
- setup.h
  - setup, [108](#)
- setup/setup.h, [108](#), [111](#)
- singlePressHandler
  - button.cpp, [4](#)
  - button.h, [6](#)
- STEPPER\_MOTOR
  - globals.h, [57](#)
- taskButtonPress
  - projectTasks.cpp, [102](#)
  - projectTasks.h, [106](#)
- taskEncoder
  - projectTasks.cpp, [103](#)
  - projectTasks.h, [106](#)
- taskEthernet
  - projectTasks.cpp, [103](#)
  - projectTasks.h, [106](#)
- taskScreenTimeout
  - projectTasks.cpp, [103](#)
- projectTasks.h, [106](#)
- taskShowTime
  - projectTasks.cpp, [103](#)
  - projectTasks.h, [107](#)
- taskStartScreen
  - projectTasks.cpp, [104](#)
  - projectTasks.h, [107](#)
- taskUpdateScreen
  - projectTasks.cpp, [104](#)
  - projectTasks.h, [107](#)
- TEXT
  - globals.h, [57](#)
- TEXT\_TYPE
  - globals.h, [56](#)
- UPDATE\_CONFIG\_ASPECT
  - globals.h, [57](#)
- UPDATE\_CONFIG\_BACKGROUND\_COLOR
  - globals.h, [57](#)
- UPDATE\_CONFIG\_BRIGHTNESS
  - globals.h, [57](#)
- UPDATE\_MAIN\_MENU
  - globals.h, [57](#)
- updateDial
  - GUI.cpp, [65](#)
  - GUI.h, [69](#)