



WHADDA WPI437 1.3 Inch OLED Screen for Arduino User Manual

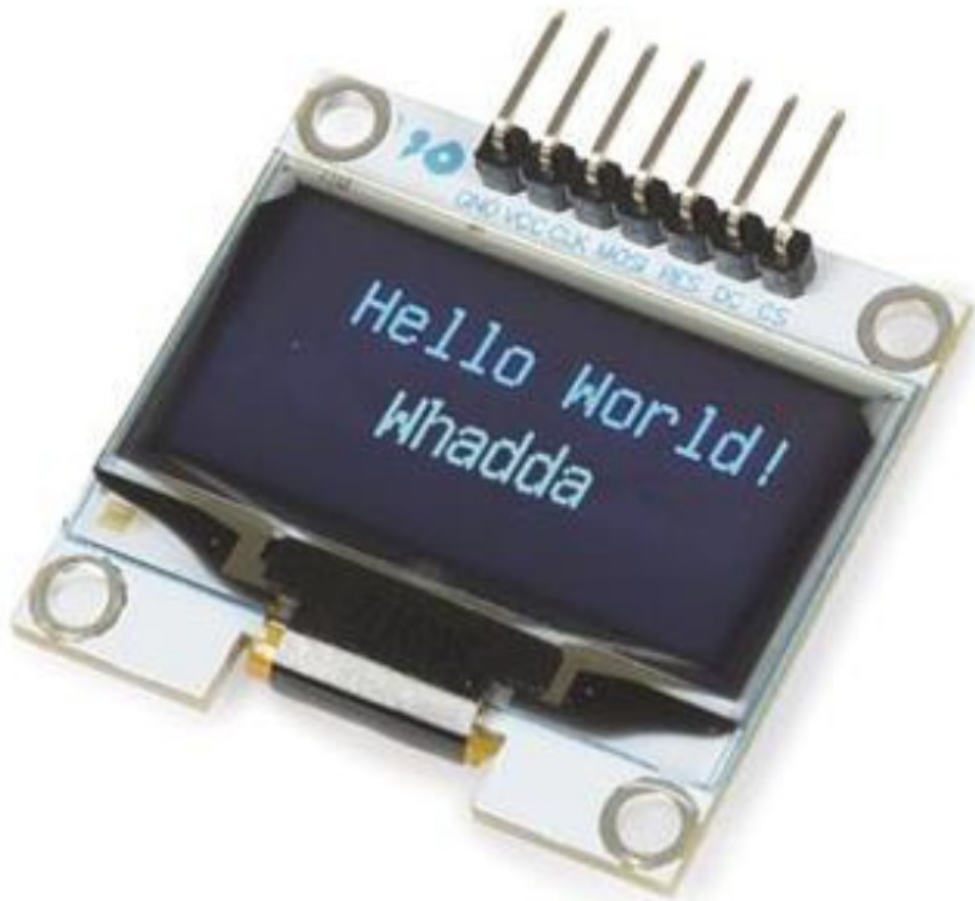
[Home](#) » [WHADDA](#) » WHADDA WPI437 1.3 Inch OLED Screen for Arduino User Manual 

Contents

- 1 WPI437 1.3 Inch OLED Screen for Arduino
- 2 Product Overview
- 3 Safety Instructions
- 4 Product Usage Instructions
- 5 Introduction
- 6 Safety Instructions
- 7 General Guidelines
- 8 What is Arduino
- 9 Product Overview
- 10 Specifications
- 11 Pin Layout
- 12 Example
- 13 Uploading the Sketch
- 14 All rights reserved
- 15 Documents / Resources
 - 15.1 References
- 16 Related Posts

WHADDA

WPI437 1.3 Inch OLED Screen for Arduino



Product Overview

The Whadda product is an OLED display that has several advantages such as low power consumption, high resolution, and a bright, large viewing angle for better readability. It requires a 2.8-5.5 V power supply and has a pin layout that includes SCK clock, MOSI data, reset, data/command, chip-select signal, and ground.

Safety Instructions

Before using the Whadda product, it is important to read and understand the instruction manual and all safety signs. The product is designed for indoor use only to ensure safe usage.

Product Usage Instructions

Before bringing the Whadda product into service, it is recommended to read the manual thoroughly. If the device was damaged during delivery, do not install or use it and contact your dealer for assistance.

To connect the Whadda product:

1. Connect VCC pin to 5V and GND pin to GND
2. Connect D0 (SCL) to D4, and RES (reset) to RESET
3. Connect D1 (SCK) to D5, CS to D6, and DC (data/command) to D7

To upload the sketch:

1. Go to Files Examples and scroll down to U8glib.
2. Open the example Graphicstest.
3. In the sketch Graphicstest, several types of displays can be selected. Un-comment U8GLIB_SH1106_128X64 u8g(4, 5, 6, 7) for WPI437.
4. Compile and upload the sketch to your WPB100 for usage.

The Graphics test sketch with only the correct driver line for WPI437 is available in the code section of the manual. The product can now be used for its intended purpose.

Introduction

To all residents of the European Union

Important environmental information about this product

This symbol on the device or the package indicates that disposal of the device after its lifecycle could harm the environment. Do not dispose of the unit (or batteries) as unsorted municipal waste; it should be taken to a specialized company for recycling. This device should be returned to your distributor or to a local recycling service. Respect the local environmental rules.

If in doubt, contact your local waste disposal authorities.

Thank you for choosing Whadda! Please read the manual thoroughly before bringing this device into service. If the device was damaged in transit, do not install or use it and contact your dealer.

Safety Instructions

- Read and understand this manual and all safety signs before using this appliance.
- For indoor use only.
- This device can be used by children aged from 8 years and above, and persons with reduced physical, sensory or mental capabilities or lack of experience and knowledge if they have been given supervision or instruction concerning the use of the device in a safe way and understand the hazards involved. Children shall not play with the device. Cleaning and user maintenance shall not be made by children without supervision.

General Guidelines

- Refer to the Velleman® Service and Quality Warranty on the last pages of this manual.
- All modifications of the device are forbidden for safety reasons. Damage caused by user modifications to the device is not covered by the warranty.
- Only use the device for its intended purpose. Using the device in an unauthorized way will void the warranty.
- Damage caused by disregard of certain guidelines in this manual is not covered by the warranty and the dealer will not accept responsibility for any ensuing defects or problems.
- Nor Velleman Group nv nor its dealers can be held responsible for any damage (extraordinary, incidental or indirect) – of any nature (financial, physical...) arising from the possession, use or failure of this product.
- Keep this manual for future reference.

What is Arduino

Arduino® is an open-source prototyping platform based on easy-to-use hardware and software. Arduino® boards are able to read inputs – light-on sensor, a finger on a button or a Twitter message – and turn it into an output –

activating of a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so, you use the Arduino programming language (based on Wiring) and the Arduino® software IDE (based on Processing). Additional shields/modules/components are required for reading a twitter message or publishing online. Surf to www.arduino.cc for more information.

Product Overview

OLED displays have several advantages: low power consumption, a bright, large viewing angle for better readability and high resolution.

Specifications

- **resolution:** 128 x 64 dots
- **viewing angle:** > 160°
- **working voltage:** 3-5 V
- **recommended library:** U8glib
- **interface:** SPI
- **driver:** SSH1106
- **working temperature:** -30 °C – 70 °C
- **OLED colour:** blue
- **dimensions:** 35 x 33.5 mm

Pin Layout

- **VDD:** 2.8-5.5 V power supply
- **SCK:** CLK clock
- **SDA:** MOSI data
- **RES:** reset
- **DC:** data/command
- **CS:** chip-select signal
- **GND:** ground

Example

Connection

- VCC → 5 V
- GND → GND
- D0 (SCL) → D4
- RES → RESET
- D1 (SCK) → D5
- CS → D6
- DC → D7

Uploading the Sketch

Go to the product page on www.velleman.eu and download the U8glib.zip file. Start the Arduino® IDE and import this library: Sketch → Include Library → Add Zip library. Once finished, go back to Sketch → Include Library → Manage libraries, and scroll down until you find the U8glib library. Select this library and tap “Update”. Now you have the latest version with examples

Go to Files → Examples and scroll down to U8glib. Open the example Graphicstest. In the sketch “Graphicstest”, several types of displays can be selected. Just “un-comment” the one you need. For the WPI437 you have to un-comment: U8GLIB_SH1106_128X64 u8g(4, 5, 6, 7); Compile and upload the sketch to your WPB100 and enjoy

Code

The “Graphicstest” sketch with only the correct driver line for WPI437 looks like this:

GraphicsTest.pde

- Before compiling: Please remove comment from the constructor of the
- connected graphics display (see below).

Universal 8bit Graphics Library, <https://github.com/olikraus/u8glib/> Copyright (c) 2012, olikraus@gmail.com

All rights reserved

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

```
#include "U8glib.h"
```

```
// setup u8g object, please remove comment from one of the following constructor calls
// IMPORTANT NOTE: The following list is incomplete. The complete list of supported
// devices with all constructor calls is here:
https://github.com/olikraus/u8glib/wiki/device
```

```
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NO_ACK); // Display which does not send
AC_WPI437 - Velleman, UN-comment this line as it is now
```

```
void u8g_prepare(void){
  u8g.setFont(u8g_font_6x10);
  u8g.setFontRefHeightExtendedText();
  u8g.setDefaultForegroundColor();
  u8g.setFontPosTop();
}
```

```
void u8g_box_frame(uint8_t a){
  u8g.drawStr(0,0,"drawBox");
  u8g.drawBox(5,10,20,10);
  u8g.drawBox(10+a,15,30,7);
  u8g.drawStr(0,30,"drawFrame");
  u8g.drawFrame(5,10+30,20,10);
  u8g.drawFrame(10+a,15+30,30,7);
}
```

```
void u8g_disc_circle(uint8_t a){
  u8g.drawStr(0,0,"drawDisc");
  u8g.drawDisc(10,18,9);
```

```
    u8g.drawDisc(24+a,16,7);
    u8g.drawStr(0,30,"drawCircle");
    u8g.drawCircle(10,18+30,9);
    u8g.drawCircle(24+a,16+30,7);
}
```

```
void u8g_r_frame(uint8_t a){
  u8g.drawStr(0,0,"drawRFrame/Box");
  u8g.drawRFrame(5,10,40,30,a+1);
  u8g.drawRBox(50,10,25,40,a+1);
}
```

```
void u8g_string(uint8_t a){
  u8g.drawStr(30+a,31,"0");
  u8g.drawStr(90(30,31+a,"90");
  u8g.drawStr(180(30-a,31,"180");
  u8g.drawStr(270(30,31-a,"270");
}
```

```
void u8g_line(uint8_t a){
  u8g.drawStr(0,0,"drawLine");
  u8g.drawLine(7+a,10,40,55);
  u8g.drawLine(7+a*2,10,60,55);
  u8g.drawLine(7+a*3,10,80,55);
  u8g.drawLine(7+a*4,10,100,55);
}
```

```
void u8g_triangle(uint8_t a){
  uint16_t offset = a;
  u8g.drawStr(0,0,"drawTriangle");
  u8g.drawTriangle(14,7,45,30,10,40);
  u8g.drawTriangle(14+offset,7+offset,45+offset,30+offset,57+offset,10+offset);
  u8g.drawTriangle(57+offset*2,10,45+offset*2,30,86+offset*2,53);
  u8g.drawTriangle(10+offset,40+offset,45+offset,30+offset,86+offset,53+offset);
}
```

```
void u8g_ascii_1(){
  char s[2]="";
  uint8_t x,y;
  u8g.drawStr(0,0,"ASCII page 1");
  for( y = 0; y < 6; y++){
    for( x = 0; x < 16; x++){
      s[0]=y*16+x+32;
      u8g.drawStr(x*7,y*10+10,s);
    }
  }
}
```

```
void u8g_ascii_2(){
  char s[2]="";
  uint8_t x,y;
  u8g.drawStr(0,0,"ASCII page 2");
  for( y = 0; y < 6; y++){
```

```

    for( x = 0; x < 16; x++ ){
        s[0] = y*16 + x + 160;
        u8g.drawStr(x*7, y*10+10, s);
    }
}

void u8g_extra_page(uint8_t a)
{
    if ( u8g.getMode() == U8G_MODE_HICOLOR || u8g.getMode() == U8G_MODE_R3G3B2 ){
        /* draw background (area is 128x128) */
        u8g_uint_t r, g, b;
        b = a << 5;
        for( g = 0; g < 64; g++ )
        {
            for( r = 0; r < 64; r++ )
            {
                u8g.setRGB(r<<2, g<<2, b);
                u8g.drawPixel(g, r);
            }
        }
        u8g.setRGB(255,255,255);
        u8g.drawStr( 66, 0, "Color Page");
    }
    else if ( u8g.getMode() == U8G_MODE_GRAY2BIT )
    {
        u8g.drawStr( 66, 0, "Gray Level");
        u8g.setColorIndex(1);
        u8g.drawBox(0, 4, 64, 32);
        u8g.drawBox(70, 20, 4, 12);
        u8g.setColorIndex(2);
        u8g.drawBox(0+1*a, 4+1*a, 64-2*a, 32-2*a);
        u8g.drawBox(74, 20, 4, 12);
        u8g.setColorIndex(3);
        u8g.drawBox(0+2*a, 4+2*a, 64-4*a, 32-4*a);
        u8g.drawBox(78, 20, 4, 12);
    }
    else
    {
        u8g.drawStr( 0, 12, "setScale2x2");
        u8g.setScale2x2();
        u8g.drawStr( 0, 6+a, "setScale2x2");
        u8g.undoScale();
    }
}

uint8_t draw_state = 0;

void draw(void){
    u8g_prepare();
    switch(draw_state >> 3){
        case 0: u8g_box_frame(draw_state&7); break;

                case 1: u8g_disc_circle(draw_state&7); break;
                case 2: u8g_r_frame(draw_state&7); break;
                case 3: u8g_string(draw_state&7); break;
                case 4: u8g_line(draw_state&7); break;
                case 5: u8g_triangle(draw_state&7); break;
                case 6: u8g_ascii_1(); break;
                case 7: u8g_ascii_2(); break;
                case 8: u8g_extra_page(draw_state&7); break;
    }
}

void setup(void){

    // flip screen, if required
    //u8g.setRot180();

    #if defined(ARDUINO)
    pinMode(13, OUTPUT);
    digitalWrite(13, HIGH);
    #endif
}


void loop(void){
    // picture loop
    u8g.firstPage();
    do {
        draw();
    } while( u8g.nextPage() );

    // increase the state
    draw_state++;
    if ( draw_state >= 9*8 )
        draw_state = 0;








    // rebuild the picture after some delay
    //delay(150);
}

```

Documents / Resources

	<p>WHADDA WPI437 1.3 Inch OLED Screen for Arduino [pdf] User Manual WPI437, WPI437 1.3 Inch OLED Screen for Arduino, 1.3 Inch OLED Screen for Arduino, OLED Screen for Arduino, Screen for Arduino</p>
---	--

References

-  [Whadda - Exciting Electronics](#)
-  [Arduino - Home](#)
-  [Arduino - Home](#)
-  [Velleman – Wholesaler and developer of electronics](#)
-  [Velleman – Wholesaler and developer of electronics](#)
-  [GitHub - olikraus/u8glib: Arduino Monochrom Graphics Library for LCDs and OLEDs](#)
-  [device · olikraus/u8glib Wiki · GitHub](#)