|  |  |
| --- | --- |
| **IMPORTANTE:**   * Verifique en cada modelo donde estan los pines VCC y GRD * FUNCIONA A 3.3 V !!!!   Todas son de ancho 128  El largo varia de 32 a 64 las grandes |  |

Agregar librerias

* Adafruit SSD1306
* Adafruit GFX

1. **Sketch**> **Include Library** > **Manage Libraries**.

2. Type “**SSD1306**” in the search box and install the SSD1306 library from Adafruit.

3. Type “**GFX**” in the search box and install the library.

* display.clearDisplay() – all pixels are off
* display.drawPixel(x,y, color) – plot a pixel in the x,y coordinates
* display.setTextSize(n) – set the font size, supports sizes from 1 to 8
* display.setCursor(x,y) – set the coordinates to start writing text
* display.print(“message”) – print the characters at location x,y
* display.display() – call this method for the changes to make effect

https://randomnerdtutorials.com/guide-for-oled-display-with-arduino/

Line

display.drawLine(0, 0, 127, 20, WHITE);

Rectangle

display.drawRect(10, 10, 50, 30, WHITE);

Fill

fillRect(x, y, width, height, color)

display.drawRoundRect(10, 10, 30, 50, 2, WHITE);

display.fillRoundRect(10, 10, 30, 50, 2, WHITE);

display.drawTriangle(10, 10, 55, 20, 5, 40, WHITE);

display.fillTriangle(10, 10, 55, 20, 5, 40, WHITE);

display.invertDisplay(true);

#include <SPI.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#si es OLED panel MAS LARGO en vez de 32 es 64

define OLED\_RESET -1 // Reset pin # or -1 if sharing Arduino reset pin)

Adafruit\_SSD1306 display(128, 32, &Wire, OLED\_RESET);

void setup()

{

// initialize with the I2C addr 0x3C

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

}

void OLED(String msg)

{

display.clearDisplay(); // Clear the buffer.

display.setTextSize(1); // Display Text

display.setTextColor(WHITE);

display.setCursor(0,5); // x,y

display.println("Hello world!");

display.display();

delay(2000);

display.clearDisplay();

display.setTextColor(BLACK, WHITE); // 'inverted' text color

display.setCursor(0,5);

display.println("Hello world!");

display.display();

delay(2000);

display.clearDisplay();

display.setTextColor(WHITE); // Changing Font Size

display.setCursor(0,5);

display.setTextSize(2); // font size

display.println("Hello!");

display.display();

delay(2000);

display.clearDisplay();

// Display Numbers

display.setTextSize(1);

display.setCursor(0,5);

display.println(123456789);

display.display();

delay(2000);

display.clearDisplay();

// Specifying Base For Numbers

display.setCursor(0,5);

display.print("0x"); display.print(0xFF, HEX);

display.print("(HEX) = ");

display.print(0xFF, DEC);

display.println("(DEC)");

display.display();

delay(2000);

display.clearDisplay();

// Display ASCII Characters

display.setCursor(0,5);

display.setTextSize(2);

display.write(3);

display.display();

delay(2000);

display.clearDisplay();

// Scroll full screen

display.setCursor(0,0);

display.setTextSize(1);

display.println("Full");

display.println("screen");

display.println("scrolling!");

display.display();

display.startscrollright(0x00, 0x07);

delay(2000);

display.stopscroll();

delay(1000);

display.startscrollleft(0x00, 0x07);

delay(2000);

display.stopscroll();

delay(1000);

display.startscrolldiagright(0x00, 0x07);

delay(2000);

display.startscrolldiagleft(0x00, 0x07);

delay(2000);

display.stopscroll();

display.clearDisplay();

// Scroll part of the screen

display.setCursor(0,0);

display.setTextSize(1);

display.println("Scroll");

display.println("some part");

display.println("of the screen.");

display.display();

display.startscrollright(0x00, 0x00);

}

void loop() {}

// =============

Displaying simple Text (Hello World)



// Display Text

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,28);

display.println("Hello world!");

display.display();

delay(2000);

Displaying Inverted Text



// Display Inverted Text

display.clearDisplay();

display.setTextColor(BLACK, WHITE); // 'inverted' text

display.setCursor(0,28);

display.println("Hello world!");

display.display();

delay(2000);

Scaling Font Size



// Changing Font Size

display.clearDisplay();

display.setTextColor(WHITE);

display.setCursor(0,24);

display.setTextSize(2);

display.println("Hello!");

display.display();

delay(2000);

Displaying Numbers



// Display Numbers

display.clearDisplay();

display.setTextSize(1);

display.setCursor(0,28);

display.println(123456789);

display.display();

delay(2000);

Specifying Base For Numbers



// Specifying Base For Numbers

display.clearDisplay();

display.setCursor(0,28);

display.print("0x"); display.print(0xFF, HEX);

display.print("(HEX) = ");

display.print(0xFF, DEC);

display.println("(DEC)");

display.display();

delay(2000);

The print() & println() functions has optional second parameter that specifies the base (format) to use; permitted values are BIN (binary, or base 2), OCT (octal, or base 8), DEC (decimal, or base 10), HEX (hexadecimal, or base 16). For floating point numbers, this parameter specifies the number of decimal places to use. For example:

* print(78, BIN) gives “1001110”
* print(78, OCT) gives “116”
* print(78, DEC) gives “78”
* print(78, HEX) gives “4E”
* println(1.23456, 0) gives “1”
* println(1.23456, 2) gives “1.23”
* println(1.23456, 4) gives “1.2346”

Displaying ASCII Symbols



// Display ASCII Characters

display.clearDisplay();

display.setCursor(0,24);

display.setTextSize(2);

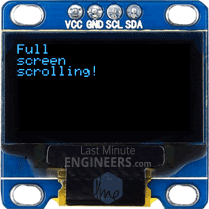
display.write(3);

display.display();

delay(2000);

The print() & println() functions send data to the display as human-readable ASCII text while write() function sends binary data to the display. So, you can use this function to display ASCII symbols. In our example sending number 3 will display heart symbol.

Full Screen Scrolling



// Scroll full screen

display.clearDisplay();

display.setCursor(0,0);

display.setTextSize(1);

display.println("Full");

display.println("screen");

display.println("scrolling!");

display.display();

display.startscrollright(0x00, 0x07);

delay(2000);

display.stopscroll();

delay(1000);

display.startscrollleft(0x00, 0x07);

delay(2000);

display.stopscroll();

delay(1000);

display.startscrolldiagright(0x00, 0x07);

delay(2000);

display.startscrolldiagleft(0x00, 0x07);

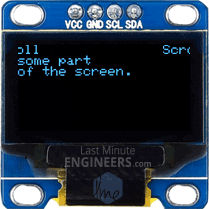
delay(2000);

display.stopscroll();

You can scroll the display horizontally by calling startscrollright() & startscrollleft() functions and diagonally by calling startscrolldiagright() & startscrolldiagleft(). All these functions accept two parameters viz. start page and stop page. Refer to OLED Memory Map section for explanation of the pages. As there are eight pages in the display from 0 to 7, you can scroll entire screen by scrolling all the pages i.e. passing parameters 0x00 and 0x07.

To stop display from scrolling you can use stopscroll() function.

Scrolling Specific Part



// Scroll part of the screen

display.setCursor(0,0);

display.setTextSize(1);

display.println("Scroll");

display.println("some part");

display.println("of the screen.");

display.display();

display.startscrollright(0x00, 0x00);

Sometimes we don’t want to scroll entire display. You can do that by passing proper start page and stop page information to scrolling functions. Refer to OLED Memory Map section for explanation of the pages. As there are eight pages in the display from 0 to 7, you can scroll some part of the screen by passing specific page numbers as parameters.

In our example, we passed both the parameters as 0x00. This will scroll only first page (first 8 rows) of the display.

Arduino Code – Basic Drawings

In this example, we’re going to try some basic drawings. This sketch demonstrates many drawing functions, including rectangles, round rectangles, circles and triangles. Try the sketch out and then we will dissect it in some detail.

#include <SPI.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

Adafruit\_SSD1306 display(-1);

void setup()

{

// initialize with the I2C addr 0x3C

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

// Clear the buffer.

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Rectangle");

display.drawRect(0, 15, 60, 40, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Filled Rectangle");

display.fillRect(0, 15, 60, 40, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Round Rectangle");

display.drawRoundRect(0, 15, 60, 40, 8, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Filled Round Rectangl");

display.fillRoundRect(0, 15, 60, 40, 8, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Circle");

display.drawCircle(20, 35, 20, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Filled Circle");

display.fillCircle(20, 35, 20, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Triangle");

display.drawTriangle(30, 15, 0, 60, 60, 60, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Filled Triangle");

display.fillTriangle(30, 15, 0, 60, 60, 60, WHITE);

display.display();

delay(2000);

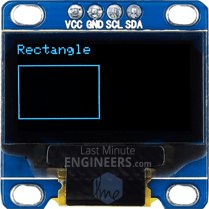
display.clearDisplay();

}

void loop() {}

Most of the code (Including libraries and initializing display) is same as above code example, except following code snippets to draw basic drawings.

Drawing Rectangle



display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Rectangle");

display.drawRect(0, 15, 60, 40, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Filled Rectangle");

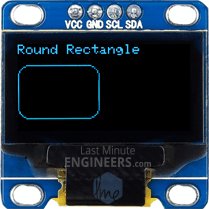
display.fillRect(0, 15, 60, 40, WHITE);

display.display();

delay(2000);

You can draw rectangle on the display by using drawRect() function. The function takes five parameters viz. X coordinate, Y coordinate, Width, Height and color. Actually this function draws hollow rectangle with 1 pixel border. You can draw filled rectangle using fillRect() function.

Drawing Round Rectangle



display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Round Rectangle");

display.drawRoundRect(0, 15, 60, 40, 8, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Filled Round Rectangl");

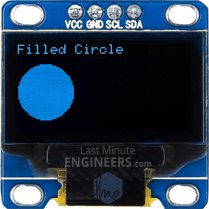
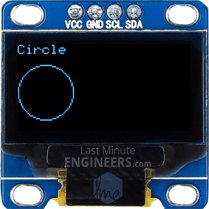
display.fillRoundRect(0, 15, 60, 40, 8, WHITE);

display.display();

delay(2000);

You can draw round rectangle on the display by using drawRoundRect() function. This function takes same parameters as drawRect() function except one additional parameter – Radius of corner rounding. Actually this function draws hollow round rectangle with 1 pixel border. You can draw filled round rectangle using fillRoundRect() function.

Drawing Circle



display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Circle");

display.drawCircle(20, 35, 20, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Filled Circle");

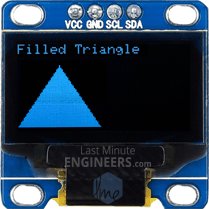
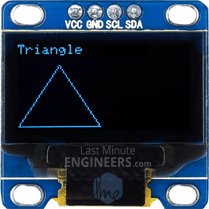
display.fillCircle(20, 35, 20, WHITE);

display.display();

delay(2000);

You can draw circle on the display by using drawCircle() function. The function takes four parameters viz. X coordinate of center, Y coordinate of center, radius and color. This function draws hollow circle with 1 pixel border. You can draw filled circle using fillCircle() function.

Drawing Triangle



display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Triangle");

display.drawTriangle(30, 15, 0, 60, 60, 60, WHITE);

display.display();

delay(2000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,0);

display.println("Filled Triangle");

display.fillTriangle(30, 15, 0, 60, 60, 60, WHITE);

display.display();

delay(2000);

You can draw triangle on the display by using drawTriangle() function. The function takes seven parameters viz. 3 X & Y coordinates (x0, y0, x1, y1, x2 & y2) of vertices of triangle and color. (X0,y0) represents top vertex, (x1,y1) represents left vertex and (x2,y2) represents right vertex.

This function draws hollow triangle with 1 pixel border. You can draw filled triangle using fillTriangle() function.

Arduino Code – Displaying Bitmap

This last example shows how to draw bitmap images to the OLED Display. This is useful for creating splash screens of company logos, making sprites or just creating fun graphics for displaying information. Copy the following code, paste it into the Arduino IDE and click upload.

#include <SPI.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

Adafruit\_SSD1306 display(-1);

// Bitmap of MarilynMonroe Image

const unsigned char MarilynMonroe [] PROGMEM = {

0xff, 0xff, 0xff, 0xff, 0xff, 0xf8, 0x1f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xc0, 0x1f, 0xff, 0xff, 0xf0, 0x41, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x80, 0x7f, 0xff, 0xff, 0xf8, 0x03, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xf9, 0xff, 0xff, 0xff, 0xe0, 0x07, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x87, 0xff, 0xff, 0xff, 0xf8, 0x03, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x07, 0xff, 0xff, 0xff, 0xf8, 0x01, 0xf1, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x9f, 0xff, 0xff, 0xff, 0xf8, 0x00, 0xf8, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xbf, 0xff, 0xff, 0xff, 0xfc, 0x02, 0x78, 0x7f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xfc, 0x3f, 0xff, 0xff, 0xfe, 0x03, 0x7c, 0x1f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xf0, 0x07, 0xff, 0xff, 0xfe, 0x01, 0xfe, 0x1f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfd, 0xe0, 0x03, 0xff, 0xff, 0xfc, 0x00, 0xfe, 0x0f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfe, 0x87, 0xe0, 0xff, 0xff, 0xfc, 0x00, 0x06, 0x07, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x1f, 0xf9, 0xff, 0xff, 0xfc, 0x00, 0x02, 0x07, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf8, 0x1f, 0xff, 0xff, 0xff, 0xfc, 0x00, 0xc3, 0xc3, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf0, 0x3f, 0xff, 0xff, 0xe0, 0x0c, 0x00, 0xe7, 0x81, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf0, 0x0f, 0xff, 0xff, 0xe0, 0x02, 0x00, 0x02, 0x00, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf0, 0x0f, 0xff, 0xff, 0xe0, 0x01, 0x00, 0x00, 0x00, 0x3f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0x80, 0x00, 0x3f, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x1e, 0x3f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xfc, 0x00, 0x00, 0x0f, 0xff, 0x3f, 0xf8, 0x00, 0x18, 0x7f, 0x1f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xf8, 0x01, 0x80, 0x03, 0xfc, 0x3f, 0xfc, 0x00, 0x70, 0xfe, 0x1f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xf0, 0x43, 0xff, 0xff, 0xf8, 0x7f, 0xf8, 0x00, 0x00, 0x7e, 0x1f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xe0, 0x07, 0xff, 0xff, 0xf0, 0xff, 0xfc, 0x00, 0x00, 0x7c, 0x3f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xe0, 0x0f, 0xff, 0xff, 0xf1, 0xef, 0xf8, 0x00, 0x01, 0xfc, 0x3f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xe4, 0xff, 0xff, 0xff, 0xf3, 0x80, 0xa0, 0x00, 0x07, 0xfc, 0xaf, 0xff, 0xff,

0xff, 0xff, 0xff, 0xec, 0x5f, 0xff, 0xff, 0xe7, 0xf0, 0x00, 0x00, 0x03, 0xfe, 0xdf, 0xff, 0xff,

0xff, 0xff, 0xff, 0xee, 0x7f, 0xff, 0xff, 0xc7, 0xf8, 0x00, 0x00, 0x03, 0xff, 0xdf, 0xff, 0xff,

0xff, 0xff, 0xff, 0xfe, 0x7f, 0xff, 0xf7, 0xc7, 0xff, 0x06, 0x00, 0x03, 0xff, 0xbf, 0xff, 0xff,

0xff, 0xff, 0xff, 0xfe, 0x5f, 0xff, 0xc7, 0x07, 0xff, 0x80, 0x00, 0x07, 0xdb, 0xbf, 0xff, 0xff,

0xff, 0xff, 0xff, 0xee, 0xff, 0xff, 0x80, 0x03, 0xff, 0xc0, 0x00, 0x03, 0xc3, 0x0f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xfe, 0xff, 0xff, 0x98, 0x03, 0xff, 0xf8, 0x00, 0x07, 0xe0, 0x0f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xef, 0xff, 0xff, 0xf8, 0x01, 0xff, 0xfc, 0x01, 0x07, 0xfc, 0x1f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xcf, 0xef, 0xff, 0xff, 0xe1, 0xff, 0xfc, 0x01, 0x07, 0xf8, 0x1f, 0xff, 0xff,

0xff, 0xff, 0xff, 0x9f, 0xff, 0xff, 0x7f, 0xf1, 0xff, 0xf8, 0x02, 0x07, 0x88, 0x3f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xcf, 0xef, 0xf8, 0x0f, 0xff, 0xff, 0xe0, 0x00, 0x07, 0x84, 0x3f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xe7, 0xef, 0xf0, 0x04, 0x7f, 0xff, 0xc0, 0x00, 0x07, 0x84, 0x7f, 0xff, 0xff,

0xff, 0xff, 0xff, 0x3f, 0xff, 0xe0, 0x00, 0x1f, 0xff, 0x80, 0x00, 0x06, 0x04, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x3f, 0x7f, 0xe1, 0xf0, 0x07, 0xff, 0x80, 0x00, 0x07, 0x06, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xc3, 0xfe, 0x03, 0xff, 0x00, 0x00, 0x03, 0x80, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xf2, 0x3f, 0xc6, 0x7f, 0x81, 0xce, 0x00, 0x00, 0x01, 0xc1, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xe0, 0x3f, 0xc0, 0x07, 0xc1, 0xfe, 0x00, 0x00, 0x0d, 0xc0, 0x7f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xe0, 0x3f, 0xc0, 0x01, 0xe0, 0xfc, 0x00, 0x00, 0x0f, 0xc0, 0x7f, 0xff, 0xff,

0xff, 0xff, 0xff, 0xc0, 0x3f, 0xc0, 0x00, 0x50, 0xfc, 0x00, 0x00, 0x0e, 0xc0, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xc0, 0x3f, 0xc0, 0x00, 0x18, 0xf8, 0x00, 0x00, 0x0e, 0xc1, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xc0, 0x3f, 0xc0, 0x00, 0x00, 0xf8, 0x00, 0x00, 0x66, 0x81, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xc0, 0x1f, 0xc7, 0x80, 0x00, 0xf8, 0x00, 0x01, 0xe0, 0x00, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xc0, 0x1f, 0xc1, 0xe0, 0x01, 0xf8, 0x00, 0x03, 0xf0, 0x01, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x80, 0x1f, 0xc0, 0x3e, 0x03, 0xf0, 0x00, 0x00, 0xe0, 0x03, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x00, 0x1f, 0xe0, 0xe0, 0x03, 0xf2, 0x00, 0x00, 0xc0, 0x03, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x80, 0x1f, 0xf0, 0x00, 0x07, 0xe6, 0x00, 0x00, 0xc0, 0x03, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x80, 0x1f, 0xff, 0x00, 0x1f, 0xee, 0x00, 0x00, 0x80, 0x07, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xb8, 0x0f, 0xff, 0xf0, 0x3f, 0xdc, 0x00, 0x00, 0x00, 0x0f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xbc, 0x0f, 0xff, 0xff, 0xff, 0xdc, 0x00, 0x00, 0x00, 0x0f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x9e, 0x0f, 0xff, 0xff, 0xff, 0xf8, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x08, 0x0f, 0xff, 0xff, 0xff, 0x70, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x00, 0x0b, 0xff, 0xff, 0xfe, 0xe0, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x00, 0x0b, 0xff, 0xff, 0xf9, 0xc0, 0x00, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x3c, 0x09, 0xff, 0xff, 0xf1, 0x80, 0x00, 0x00, 0x00, 0x7f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x1e, 0x08, 0x3f, 0xff, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x1f, 0x08, 0x03, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0x80, 0x1c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xce, 0x1c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xfe, 0x1c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0x7e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xff, 0xff, 0xff

};

void setup()

{

// initialize with the I2C addr 0x3C

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

// Clear the buffer.

display.clearDisplay();

// Display bitmap

display.drawBitmap(0, 0, MarilynMonroe, 128, 64, WHITE);

display.display();

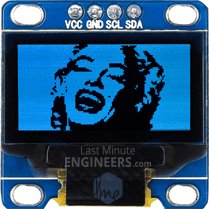
// Invert Display

//display.invertDisplay(1);

}

void loop() {}

This is how the output looks like.



To show bitmap image on the OLED display we need to call drawBitmap() function. It takes six parameters viz. Top left corner X coordinate, top left corner Y coordinate, byte array of monochrome bitmap, width of bitmap in pixels, height of bitmap in pixels and Color.

In our example, the bitmap image is 128×64 in size. So, X & Y coordinates are set to 0 while width & height is set to 128 & 64.

// Display bitmap

display.drawBitmap(0, 0, MarilynMonroe, 128, 64, WHITE);

display.display();

But, before we can call the drawBitmap() function, we first need an image to draw. Remember, the screen resolution of the OLED display is 128×64 pixels, so images larger than that will not display correctly. To get a correctly sized image, you can use your favorite drawing programs like Inkscape, Photoshop, Paint, etc., setting the canvas size to 128×64 pixels.

We took laughing Marilyn Monroe image as an example and converted into 128×64 pixels using Paint and saved as .bmp.





Once you have a bitmap, it’s time to convert it into an array that the SSD1306 OLED controller can understand. This can be done using two ways: Online method using image2cpp and Offline method using LCD Assistant.

Online Bitmap Array Generator – image2cpp

There’s an online application called image2cpp – <http://javl.github.io/image2cpp/> which can convert your image into an array. Image2cpp is newer and much more powerful than LCD Assistant (later solution). It will allow you to:

* Convert multiple images simultaneously.
* Scale your image file – Stretch/Scale to fit/Original
* Adjust the Brightness threshold between black and white.
* Re-center the image vertically and / or horizontally.
* Reverse image colors

This tool is so powerful that it can work offline as well. Simply save the page to your PC and open it in your browser. Thanks to [Jasper van Loenen](https://github.com/javl) for his excellent contribution.

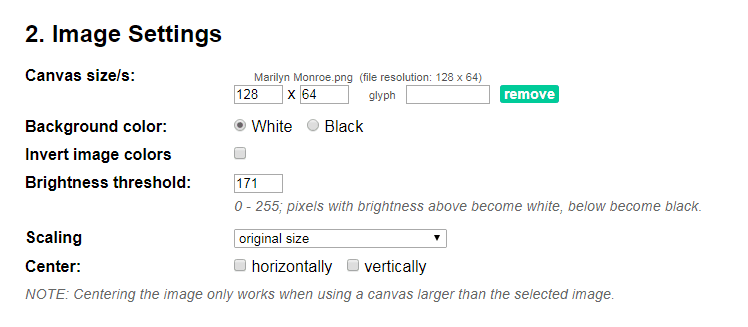
To begin with, open [image2cpp](http://javl.github.io/image2cpp/) in your browser and select any image you want to display on OLED screen.



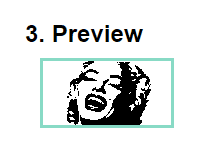
The dimensions of your image will populate in Canvas size option under Image settings. If you have selected bigger image than 128×64, change it to 128×64 and select proper Scaling option. You can view the output in Preview section.

You can change the Background color or Invert image colors if necessary.

Finally, change the most important option – Brightness threshold as per your requirement. Setting threshold will make pixels above this level white and below black. In our case we have set it to 171 to get nice details of Marilyn Monroe.

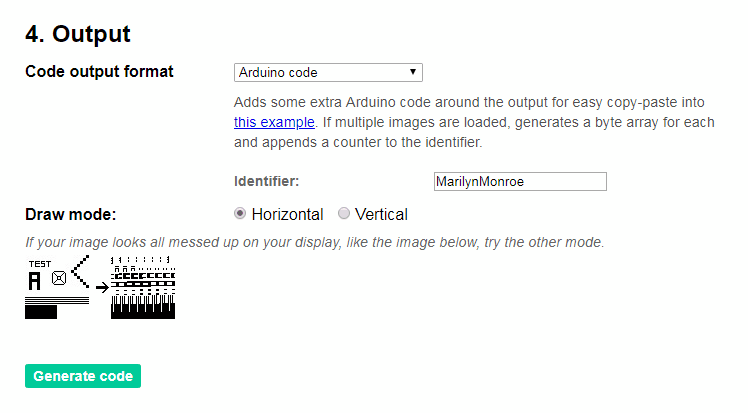


This little preview reflects whatever changes you make in your settings. You can change settings while keeping eye on it.

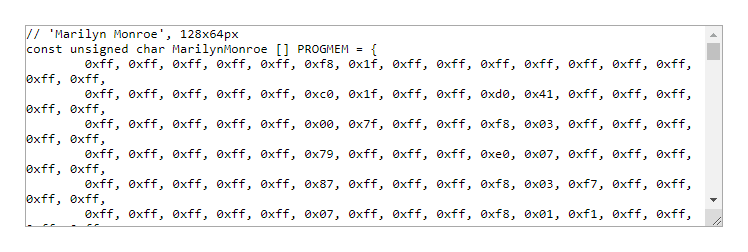


Once you are satisfied with the outcome, you can proceed generating the data array. Simply select Code output format as Arduino Code and click on Generate code button.

Just for your information, there’s an option called Draw mode. It actually creates image according to the scanning patter of the display. If your image looks all messed up on your display, try changing the mode.



That’s it. The byte array of your bitmap will be generated. You can use the output directly with our example code. Just be sure to name it appropriately. Then call your array inside the drawBitmap() function.

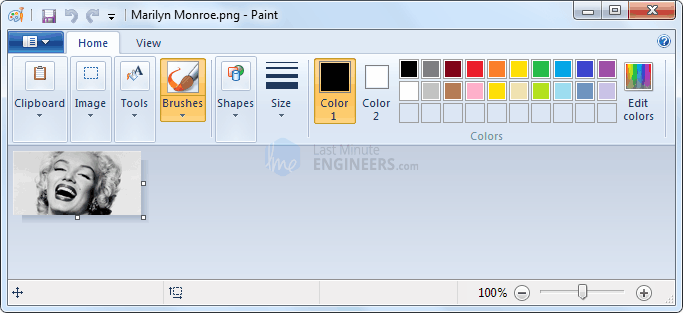


Offline Bitmap Array Generator – LCD Assistant

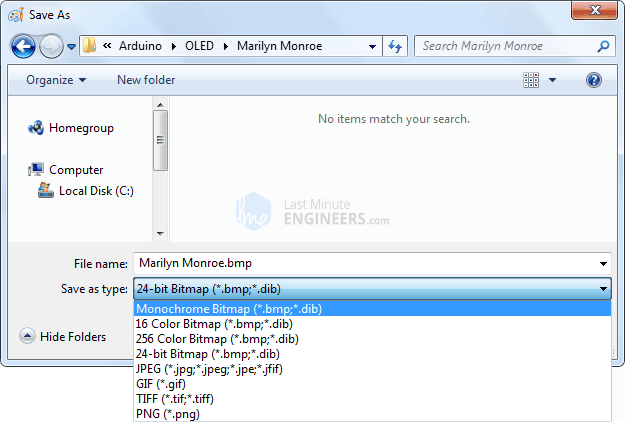
There’s another application called LCD assistant – <http://en.radzio.dxp.pl/bitmap_converter/> which can convert your bitmap image into data array. It’s not as powerful as image2cpp but still popular among hobbyists.

To start with, you need to convert you image into 128×64 1-bit monochrome bitmap. You can use your favorite drawing programs like Inkscape, Photoshop, Paint, etc. to do it, just like we did in MS paint.

Open your file in MS Paint and resize it to 128×64.

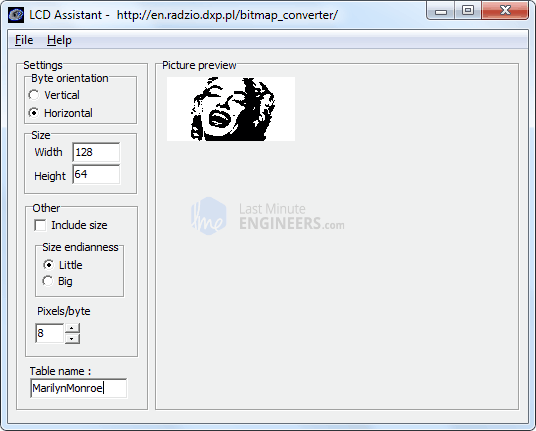


Now, save your file as bitmap. While saving the file choose Save as type : Monochrome Bitmap(\*.bmp;\*.dib). This will generate 1-bit/binary bitmap image that has only two possible values for each pixel i.e. 0 (black) or 1 (white).



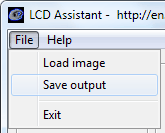
The only downside here is that you cannot set brightness threshold level. It is set to 50% by default and cannot be changed.

Anyways now, download LCD assistant program. Open the executable and load your bitmap from File menu.

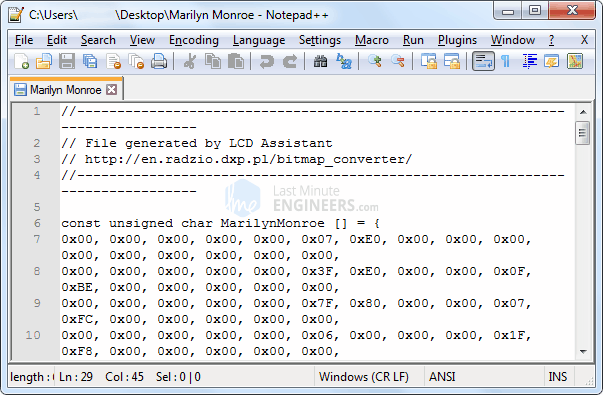


There’s nothing much you can do with this tool. So, just go to File menu and click on Save output option. Save the file as text file.

Just for your information, there’s an option called Byte Orientation. It actually creates image according to the scanning patter of the display. If your image looks all messed up on your display, try changing the mode.



That’s it. With your array created, paste it into your code.



Just be sure to name it appropriately. Then call your array inside the drawBitmap() function.