Arduino und OLED Display

Quelle: https://funduino.de/nr-42-oled-display-ssd1306-128x64-128x32

In dieser Anleitung möchten wir lernen, wie wir ein kleines **OLED Display** mit 128×64 oder 128×32 Pixeln mit Hilfe eines **Arduino** Mikrocontrollers ansteuern können.

Grundlagen

Ein Zweizeilen LCD Display 16x2 an den Arduino i2c Bus anschließen.

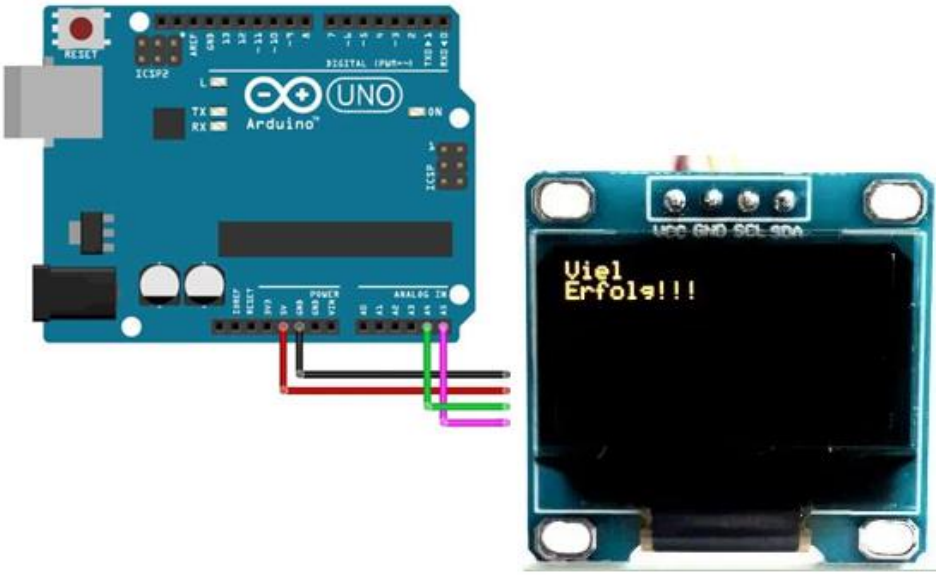
- Software-Bibliothek
- Spannungsversorgung
- Mit i2c Bus OLED 128x64 oder OLED 128x32

OLED Displays haben im Vergleich zu LCD Displays (Liquid Crystal Display – Flüssigkristallanzeige) den Vorteil, dass Sie sehr dünn sind und einen guten Kontrast aufweisen. Daher ist diese Technik in vielen Geräten des Alltags immer häufiger zu finden. Die kleinen OLED Displays im Bereich des Mikrocontrollings sind in der Regel monochrome, also einfarbige, Displays. Die leuchtenden Pixel haben im Vergleich zur Hintergrundfarbe nur eine Farbmöglichkeit. Selbst zweifarbige OLEDs, die wir z.B. in dieser Anleitung verwenden sind monochrom. Im oberen Bereich des Displays (Reihe 1-16) können die Pixel nur gelb und unten (Reihe 17-64) nur blau leuchten.

OLED Displays gibt es in vielen Varianten. Sie unterscheiden sich in Farbe, Größe und auch in der Technik der Ansteuerung. Für diese Anleitung nutzen wir ein Display mit einer Auflösung von 128×64 Pixel und einer Ansteuerung über den I²C Bus (im Deutschen gesprochen als „I-Quadrat-C“). Das OLED Display befindet sich bei diesem Modul auf einer PCB, welches den Treiberchip SSD1306 und die Anschlusspins enthält.

Praxis-tipp: Mit etwas Geschick lässt sich das eigentliche Display von der Träger-PCB trennen, wenn man mit einem Stück Papier zwischen PCB und Glas entlang das dort befindliche doppelseitige Klebeband löst. Dadurch ist das Display nur noch ca. 1mm dick und kann in noch kompakteren Anwendungen eingesetzt werden.

Schaltung Aufbau



I2C Character LCD	Arduino
GND	GND
VCC	5 V
SDA	A4
SCL	A5

I2c-Scanner

Um die i2c Bus Adresse des Bauteils zu finden verwenden wir eine Software:
I2c-scanner.ino

I2C address scanner Serial Monitor output

Library SSD1306Ascii

Für die Programmierung von OLED Displays gibt es viele Möglichkeiten. Zunächst muss überlegt werden, welche Library man verwenden möchte. Alle Libraries haben Vor- und Nachteile. Alle haben die Gemeinsamkeit, dass durch die Installation der Library auch diverse Beispielsketche mitinstalliert werden, die in der Arduino Software unmittelbar nach der Installation im Menüpunkt Datei → Beispiele auftauchen. Diese Beispiele geben viele Hinweise auf die Möglichkeiten der Programmierung und zeigen auch grafische Anwendungen auf. Jedoch bringen die vielen Möglichkeiten auch den Nachteil mit sich, dass sehr viel interner Speicher des Mikrocontrollers belegt wird. So gibt es Beispielsketche für OLED Displays, die mehr als 96% des Speicherplatzes (Arduino UNO) nutzen. Für konkrete Operationen wie dem Auswerten von Sensordaten bleibt dann zu wenig Speicherplatz übrig. Aus diesem Grund befassen wir uns in dieser Anleitung mit einer Bibliothek, die speicheroptimiert und leicht zu bedienen ist. Die Library findet man über den Bibliotheksverwalter unter dem Namen „SSD1306Ascii“.

Um einen ersten Eindruck über die Möglichkeiten zu bekommen, bietet es sich an, zunächst einige Beispielsketche zu testen, die automatisch mit der Installation der Library installiert werden. Ein Sketch heißt „FontSamplesWire“. Man findet ihn in der Arduino Software unter dem Menüpunkt Datei → Beispiele → SSD1306Ascii → FontSamplesWire. Nach dem Hochladen tauchen auf dem OLED Display viele verschiedene Schriftarten auf, die in der Library hinterlegt sind. Auf dem UNO Controller nimmt dieser Sketch 96% des Speicherplatzes ein. Hier wird also auch klar, dass während der regulären Verwendung möglichst wenig verschiedene Schriften verwendet werden sollten.

Die Schriftarten erscheinen bei dem genannten Sketch der Reihe nach auf dem OLED Display.

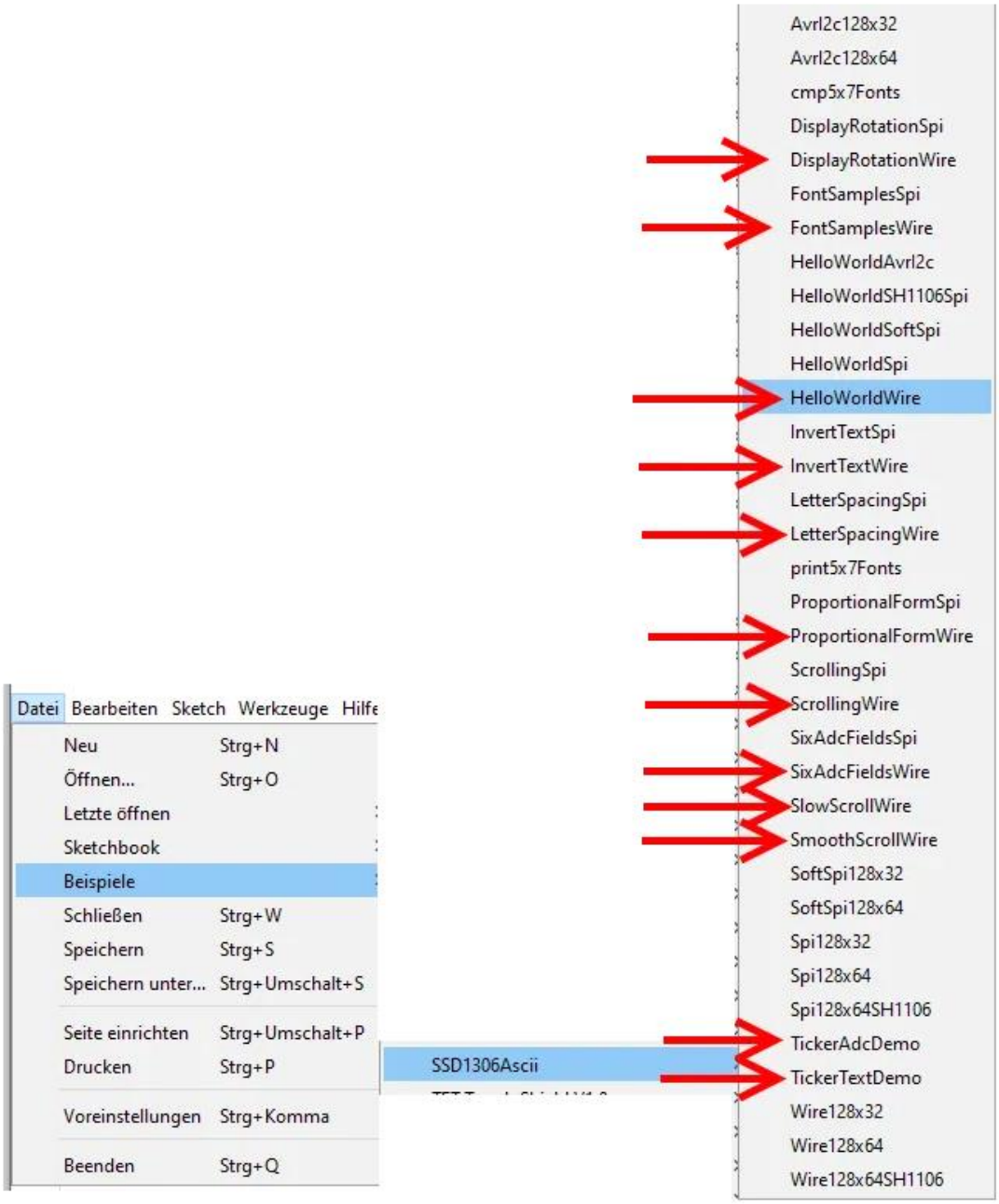
Software/Programm

Arduino_OLED.ino

```
// Beispiel-Sketch zu Arduino und OLED
// Sept 2020
// Vorlage: funduino.de/nr-42-oled-display-ssd1306-128x64-128x32

#include <Wire.h>
#include "SSD1306Ascii.h"
#include "SSD1306AsciiWire.h"
#define I2C_ADDRESS 0x3C

SSD1306AsciiWire oled;

void setup() {
  Wire.begin();
  Wire.setClock(400000L);
  oled.begin(&Adafruit128x64, I2C_ADDRESS);
}

void loop()
{
  oled.setFont(System5x7); // Auswahl der Schriftart
  oled.clear(); //Löschen der aktuellen Displayanzeige
  oled.println("Viel"); //Text in der ersten Zeile. "Println" sorgt dabei für einen Zeilensprung.
  oled.print("Erfolg!!!"); // Text in der zweiten Zeile. Da es keine dritte Zeile gibt, wird hier
  delay (2000);
}
```

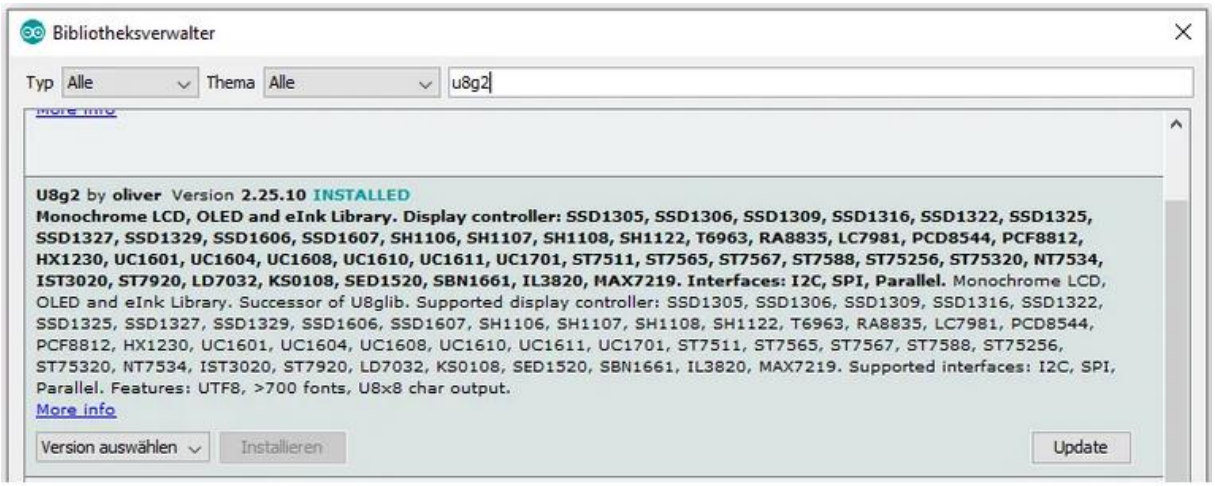
Beispiele aus der Library

Darüber hinaus gibt es selbst in dieser „kleinen“ Library viele weitere Funktionen, wie zum Beispiel Seitliches schrollen, dauerhaftes Auf- und Abscrollen, Display um 180° drehen (oled.displayRemap(true);), folgende Textzeile invertieren (oled.setInvertMode(i%2);), gesamtes Display invertieren (oled.invertDisplay(!(i%2));), Buchstabenabstand vergrößern (oled.setLetterSpacing(2);) automatisches Scrollen am Ende des Bildschirms (oled.setScrollMode(SCROLL_MODE_AUTO);) und vieles mehr. Für jede Funktion beinhaltet die Library einzelne Beispielsketche, die im folgenden Bild markiert wurden.



Library U8g2

Eine sehr Umfangreiche Library trägt den Namen „U8g2“ und ist in der Lage eine Vielzahl von verfügbaren Displays bzw. Displaycontroller anzusteuern.



Auch die enthaltenen Beispiele sind sehr umfangreich und geben insbesondere für grafische Anwendungen Hilfestellungen. Um mit der „U8g2“ Library arbeiten zu können, ist es in jedem Beispielsketch notwendig, das richtige Display zu aktivieren. Dafür muss man wissen, über welchen Displaytreiber das verwendete Display verfügt.
In unserem Fall müssen wir von den „auskommentierten“ Displays das folgende durch das Entfernen der zwei Schrägstriche „//“ aktivieren:

Bei OLED Displays mit dem SSD1306 Chip:

```
U8G2_SSD1306_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, /* reset=*/ U8X8_PIN_NONE);
```

Bei OLED Displays mit dem SH1106 Chip:

```
U8G2_SH1106_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, /* reset=*/ U8X8_PIN_NONE);
```

Library Adafruit_SSD1306

Die Bibliothek unter *https://github.com/adafruit/Adafruit_SSD1306* laden und einbinden.

Um das Display anzusteuern, gibt es nun die folgenden Befehle:

display.clearDisplay() – Display wird vollständig bereinigt
display.drawPixel(x,y, color) – Zeichnet einen Pixel auf die x,y-Koordinaten
display.setTextSize(n) – Schriftgröße festlegen. Von 1 bis 8. (bei einem zweizeiligen Display natürlich nicht mehr wie 2.
display.setCursor(x,y) – Setzt den Cursor an die x,y-Position, an der der Text beginnen soll
display.print("message") – zeigt den Text an den zuvor festgelegten Koordinaten
display.display() – Das wird immer am Ende der Befehlskette aufgerufen, damit die vorher aufgebaute/zusammengesetzte Text/Grafik-Ausgabe zum Display zur Ausgabe gesendet wird.

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 32 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
#define OLED_RESET     4 // Reset pin # (or -1 if sharing Arduino reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

void setup() {
  Serial.begin(115200);

  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // Address 0x3C for 128x64
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }
  delay(2000);
  display.clearDisplay();

  display.setTextSize(1);
  display.setCursor(0, 10);
  display.println("www.kollino.de"); //Textzeile ausgeben
  display.display();
}

void loop() {

}
```

Eine Kleinigkeit gilt es noch zu beachten bei der Verwendung von Displays: Ein Display zeigt immer das letzte an, dass ihm befohlen wurde darzustellen. So lange, wie es genügend Strom bekommt. Das heißt, man muss immer, wenn man neue Informationen anzeigen lassen will, den alten Inhalt zuerst löschen (display.clearDisplay();) und neu schreiben, sonst überschreibt man den vorhandenen Inhalt mit dem neuen Inhalt und hat nur noch Kraut und Rüben auf dem Display.

Quelle: https://www.kollino.de/arduino/oled-display-mit-ssd1306-chipsatz-via-i2c-an-arduino-anschliessen/