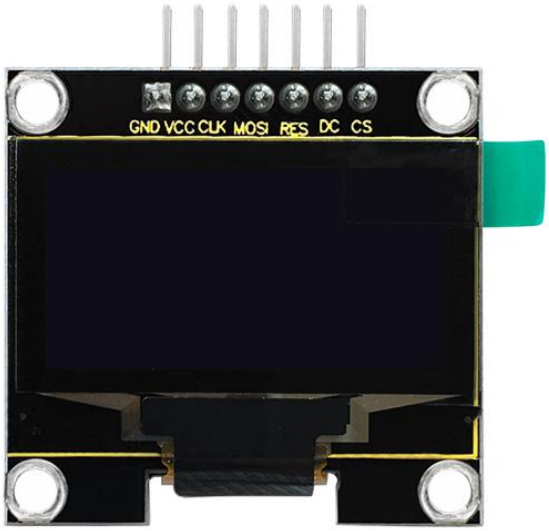
1.3" 128x64 OLED Graphic Display



Introduction

OLED is short for organic light emitting diode.

On the microscopic level, an OLED display is a matrix of organic LEDs that light up when they emit energy.

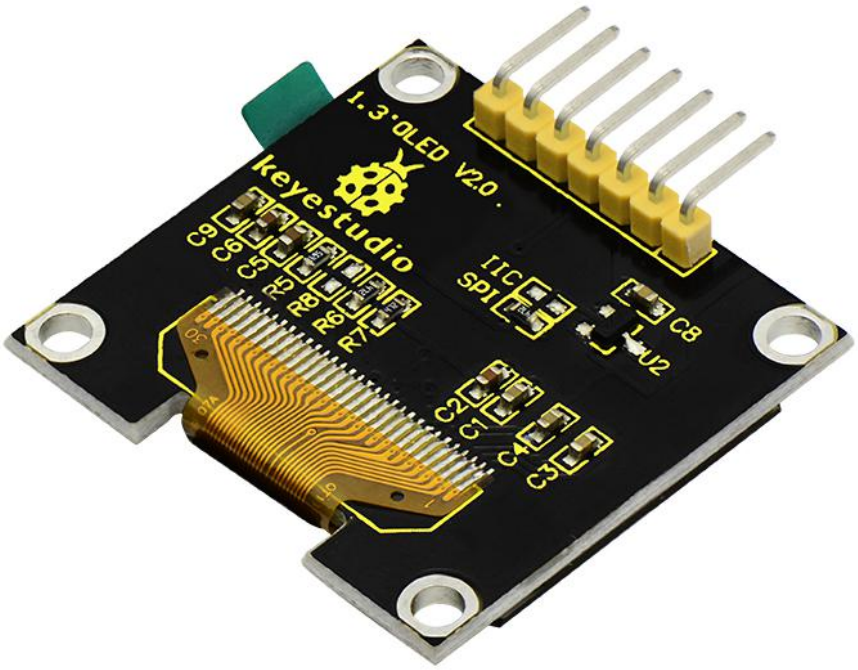
Old LCD (Liquid Crystal Display) technology uses electronically controlled polarizers to change the way light passes or does not pass through them. This requires an external backlight that lights up the whole display underneath. This uses a lot of energy because at the time the display is on, enough light for all pixels must be provided.

The new OLED technology only uses electricity per pixel. Because each pixel creates its own light, only the pixels that are on use electricity.

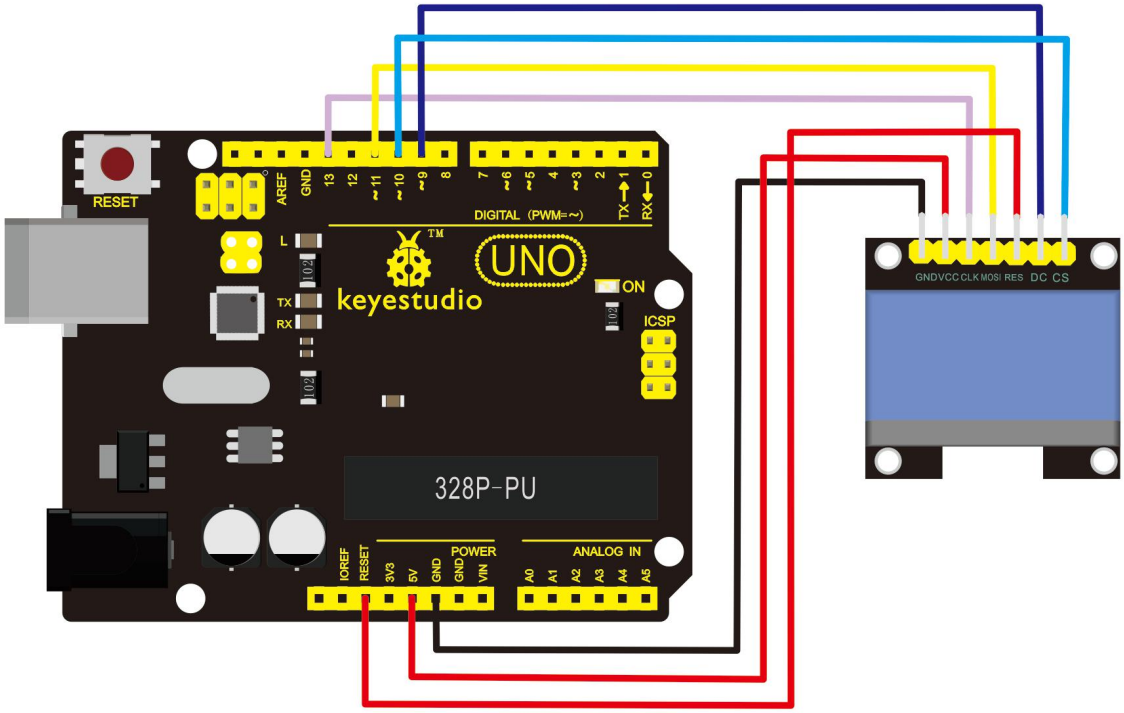
This makes OLED technology very efficient; also, the way these types of OLEDs are built allows them to be very thin compared to LCD.

Specification：

- Number of Pixels：128 × 64
- Color Depth：Monochrome (blue)
- Brightness (cd/m2)：100 (Typ) @ 12V



Connection Diagram



Sample Code

****

#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:http://code.google.com/p/u8glib/

//U8GLIB_NHD27OLED_BW u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_NHD27OLED_2X_BW u8g(13, 11, 10, 9); // SPI Com: SCK

= 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_NHD27OLED_GR u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_NHD27OLED_2X_GR u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_NHD31OLED_BW u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_NHD31OLED_2X_BW u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_NHD31OLED_GR u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_NHD31OLED_2X_GR u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_DOGS102 u8g(13, 11, 10, 9, 8); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_DOGM132 u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_DOGM128 u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_DOGM128_2X u8g(13, 11, 10, 9); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_ST7920_128X64_1X u8g(8, 9, 10, 11, 4, 5, 6, 7, 18, 17, 16);

```
// 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18, di=17,rw=16
//U8GLIB_ST7920_128X64_4X u8g(8, 9, 10, 11, 4, 5, 6, 7, 18, 17, 16);
// 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18, di=17,rw=16
//U8GLIB_ST7920_128X64_1X u8g(18, 16, 17);    // SPI Com: SCK =
en = 18, MOSI = rw = 16, CS = di = 17
//U8GLIB_ST7920_128X64_4X u8g(18, 16, 17);    // SPI Com: SCK =
en = 18, MOSI = rw = 16, CS = di = 17
//U8GLIB_ST7920_192X32_1X u8g(8, 9, 10, 11, 4, 5, 6, 7, 18, 17, 16);
// 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18, di=17,rw=16
//U8GLIB_ST7920_192X32_4X u8g(8, 9, 10, 11, 4, 5, 6, 7, 18, 17, 16);
// 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18, di=17,rw=16
//U8GLIB_ST7920_192X32_1X u8g(18, 16, 17);    // SPI Com: SCK =
en = 18, MOSI = rw = 16, CS = di = 17
//U8GLIB_ST7920_192X32_4X u8g(18, 16, 17);    // SPI Com: SCK =
en = 18, MOSI = rw = 16, CS = di = 17
//U8GLIB_ST7920_192X32_1X u8g(13, 11, 10);    // SPI Com: SCK =
en = 13, MOSI = rw = 11, CS = di = 10
//U8GLIB_ST7920_192X32_4X u8g(10);          // SPI Com: SCK = en =
13, MOSI = rw = 11, CS = di = 10, HW SPI
//U8GLIB_ST7920_202X32_1X u8g(8, 9, 10, 11, 4, 5, 6, 7, 18, 17, 16);
// 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18, di=17,rw=16
//U8GLIB_ST7920_202X32_4X u8g(8, 9, 10, 11, 4, 5, 6, 7, 18, 17, 16);
```

```
// 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18, di=17,rw=16
//U8GLIB_ST7920_202X32_1X u8g(18, 16, 17);    // SPI Com: SCK =
en = 18, MOSI = rw = 16, CS = di = 17
//U8GLIB_ST7920_202X32_4X u8g(18, 16, 17);    // SPI Com: SCK =
en = 18, MOSI = rw = 16, CS = di = 17
//U8GLIB_LM6059 u8g(13, 11, 10, 9);         // SPI Com: SCK = 13,
MOSI = 11, CS = 10, A0 = 9
//U8GLIB_LM6063 u8g(13, 11, 10, 9);         // SPI Com: SCK = 13,
MOSI = 11, CS = 10, A0 = 9
//U8GLIB_DOGXL160_BW u8g(10, 9);       // SPI Com: SCK = 13,
MOSI = 11, CS = 10, A0 = 9
//U8GLIB_DOGXL160_GR u8g(13, 11, 10, 9); // SPI Com: SCK = 13,
MOSI = 11, CS = 10, A0 = 9
//U8GLIB_DOGXL160_2X_BW u8g(13, 11, 10, 9);      // SPI Com: SCK
= 13, MOSI = 11, CS = 10, A0 = 9
//U8GLIB_DOGXL160_2X_GR u8g(13, 11, 10, 9);// SPI Com: SCK =
13, MOSI = 11, CS = 10, A0 = 9
//U8GLIB_PCD8544 u8g(13, 11, 10, 9, 8);         // SPI Com: SCK = 13,
MOSI = 11, CS = 10, A0 = 9, Reset = 8
//U8GLIB_PCF8812 u8g(13, 11, 10, 9, 8);         // SPI Com: SCK = 13,
MOSI = 11, CS = 10, A0 = 9, Reset = 8
//U8GLIB_KS0108_128 u8g(8, 9, 10, 11, 4, 5, 6, 7, 18, 14, 15, 17, 16);
```

```
        // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18, cs1=14,
cs2=15,di=17,rw=16
//U8GLIB_LC7981_160X80 u8g(8, 9, 10, 11, 4, 5, 6, 7,    18, 14, 15, 17,
16);     // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18,
cs=14 ,di=15,rw=17, reset = 16
//U8GLIB_LC7981_240X64 u8g(8, 9, 10, 11, 4, 5, 6, 7,    18, 14, 15, 17,
16);     // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18,
cs=14 ,di=15,rw=17, reset = 16
//U8GLIB_LC7981_240X128 u8g(8, 9, 10, 11, 4, 5, 6, 7,    18, 14, 15, 17,
16);     // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 en=18,
cs=14 ,di=15,rw=17, reset = 16
//U8GLIB_ILI9325D_320x240 u8g(18,17,19,U8G_PIN_NONE,16 );
            // 8Bit Com: D0..D7: 0,1,2,3,4,5,6,7 en=wr=18, cs=17, rs=19,
rd=U8G_PIN_NONE, reset = 16
//U8GLIB_SBN1661_122X32 u8g(8,9,10,11,4,5,6,7,14,15, 17,
U8G_PIN_NONE, 16);    // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7 cs1=14,
cs2=15,di=17,rw=16,reset = 16
//U8GLIB_SSD1306_128X64 u8g(13, 11, 10, 9);    // SW SPI Com: SCK
= 13, MOSI = 11, CS = 10, A0 = 9
//U8GLIB_SSD1306_128X64 u8g(4, 5, 6, 7);    // SW SPI Com: SCK =
4, MOSI = 5, CS = 6, A0 = 7 (new white HalTec OLED)
//U8GLIB_SSD1306_128X64 u8g(10, 9);          // HW SPI Com: CS = 10,
```

A0 = 9 (Hardware Pins are SCK = 13 and MOSI = 11)

//U8GLIB_SSD1306_128X64

u8g(U8G_I2C_OPT_NONE|U8G_I2C_OPT_DEV_0); // I2C / TWI

//U8GLIB_SSD1306_128X64

u8g(U8G_I2C_OPT_DEV_0|U8G_I2C_OPT_NO_ACK|U8G_I2C_OPT

_FAST); // Fast I2C / TWI

//U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NO_ACK); //

Display which does not send AC

//U8GLIB_SSD1306_ADAFRUIT_128X64 u8g(13, 11, 10, 9);// SW SPI

Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_SSD1306_ADAFRUIT_128X64 u8g(10, 9); // HW SPI

Com: CS = 10, A0 = 9 (Hardware Pins are SCK = 13 and MOSI = 11)

//U8GLIB_SSD1306_128X32 u8g(13, 11, 10, 9); // SW SPI Com: SCK

= 13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_SSD1306_128X32 u8g(10, 9); // HW SPI

Com: CS = 10, A0 = 9 (Hardware Pins are SCK = 13 and MOSI = 11)

//U8GLIB_SSD1306_128X32 u8g(U8G_I2C_OPT_NONE); // I2C /

TWI

U8GLIB_SH1106_128X64 u8g(13, 11, 10, 9); // SW SPI Com: SCK =

13, MOSI = 11, CS = 10, A0 = 9

//U8GLIB_SH1106_128X64 u8g(4, 5, 6, 7); // SW SPI Com: SCK = 4,

MOSI = 5, CS = 6, A0 = 7 (new blue HalTec OLED)

```
//U8GLIB_SH1106_128X64 u8g(U8G_I2C_OPT_NONE); // I2C / TWI
//U8GLIB_SH1106_128X64
u8g(U8G_I2C_OPT_DEV_0|U8G_I2C_OPT_FAST);   // Dev 0, Fast I2C
/ TWI
//U8GLIB_SH1106_128X64 u8g(U8G_I2C_OPT_NO_ACK); // Display
which does not send ACK
//U8GLIB_SSD1309_128X64 u8g(13, 11, 10, 9);     // SPI Com: SCK =
13, MOSI = 11, CS = 10, A0 = 9
//U8GLIB_SSD1327_96X96_GR u8g(U8G_I2C_OPT_NONE);     // I2C
//U8GLIB_SSD1327_96X96_2X_GR u8g(U8G_I2C_OPT_NONE);    //
I2C
//U8GLIB_UC1611_DOGM240 u8g(U8G_I2C_OPT_NONE); // I2C
//U8GLIB_UC1611_DOGM240 u8g(13, 11, 10, 9); // SW SPI Com: SCK
= 13, MOSI = 11, CS = 10, A0 = 9
//U8GLIB_UC1611_DOGM240 u8g(10, 9);       // HW SPI Com: CS = 10,
A0 = 9 (Hardware Pins are    SCK = 13 and MOSI = 11)
//U8GLIB_UC1611_DOGXL240 u8g(U8G_I2C_OPT_NONE);     // I2C
//U8GLIB_UC1611_DOGXL240 u8g(13, 11, 10, 9);     // SW SPI Com:
SCK = 13, MOSI = 11, CS = 10, A0 = 9
//U8GLIB_UC1611_DOGXL240 u8g(10, 9);          // HW SPI Com: CS
= 10, A0 = 9 (Hardware Pins are    SCK = 13 and MOSI = 11)
//U8GLIB_NHD_C12864 u8g(13, 11, 10, 9, 8);  // SPI Com: SCK = 13,
```

MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_NHD_C12832 u8g(13, 11, 10, 9, 8); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_LD7032_60x32 u8g(13, 11, 10, 9, 8); // SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_LD7032_60x32 u8g(11, 12, 9, 10, 8); // SPI Com: SCK = 11, MOSI = 12, CS = 9, A0 = 10, RST = 8 (SW SPI Nano Board)

//U8GLIB_UC1608_240X64 u8g(13, 11, 10, 9, 8); // SW SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_UC1608_240X64_2X u8g(13, 11, 10, 9, 8); // SW SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_UC1608_240X64 u8g(10, 9, 8); // HW SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_UC1608_240X64_2X u8g(10, 9, 8); // HW SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_UC1608_240X u8g(13, 11, 10, 9, 8);// SW SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_UC1608_240X64_2X u8g(13, 11, 10, 9, 8); // SW SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_UC1608_240X64 u8g(10, 9, 8); // HW SPI Com: SCK = 13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_UC1608_240X64_2X u8g(10, 9, 8); // HW SPI Com: SCK =

13, MOSI = 11, CS = 10, A0 = 9, RST = 8

//U8GLIB_T6963_240X128 u8g(8, 9, 10, 11, 4, 5, 6, 7, 14, 15, 17, 18, 16); // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7, cs=14, a0=15, wr=17, rd=18, reset=16

//U8GLIB_T6963_128X128 u8g(8, 9, 10, 11, 4, 5, 6, 7, 14, 15, 17, 18, 16); // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7, cs=14, a0=15, wr=17, rd=18, reset=16

//U8GLIB_T6963_240X64 u8g(8, 9, 10, 11, 4, 5, 6, 7, 14, 15, 17, 18, 16); // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7, cs=14, a0=15, wr=17, rd=18, reset=16

//U8GLIB_T6963_128X64 u8g(8, 9, 10, 11, 4, 5, 6, 7, 14, 15, 17, 18, 16); // 8Bit Com: D0..D7: 8,9,10,11,4,5,6,7, cs=14, a0=15, wr=17, rd=18, reset=16

//U8GLIB_HT1632_24X16 u8g(3, 2, 4); // WR = 3, DATA = 2, CS = 4

//U8GLIB_SSD1351_128X128_332 u8g(13, 11, 8, 9, 7); // Arduino UNO: SW SPI Com: SCK = 13, MOSI = 11, CS = 8, A0 = 9, RESET = 7 (http://electronics.ilsoft.co.uk/ArduinoShield.aspx)

//U8GLIB_SSD1351_128X128_332 u8g(76, 75, 8, 9, 7); // Arduino DUE: SW SPI Com: SCK = 13, MOSI = 11, CS = 8, A0 = 9, RESET = 7 (http://electronics.ilsoft.co.uk/ArduinoShield.aspx)

//U8GLIB_SSD1351_128X128_332 u8g(8, 9, 7); // Arduino: HW SPI

```
Com: SCK = 13, MOSI = 11, CS = 8, A0 = 9, RESET = 7
(http://electronics.ilsoft.co.uk/ArduinoShield.aspx)
//U8GLIB_SSD1351_128X128_HICOLOR u8g(76, 75, 8, 9, 7); //
Arduino DUE, SW SPI Com: SCK = 76, MOSI = 75, CS = 8, A0 = 9,
RESET = 7 (http://electronics.ilsoft.co.uk/ArduinoShield.aspx)
//U8GLIB_SSD1351_128X128_HICOLOR u8g(8, 9, 7); // Arduino, HW
SPI Com: SCK = 76, MOSI = 75, CS = 8, A0 = 9, RESET = 7
(http://electronics.ilsoft.co.uk/ArduinoShield.aspx)
//U8GLIB_SSD1351_128X128GH_332 u8g(8, 9, 7); // Arduino, HW SPI
Com: SCK = 76, MOSI = 75, CS = 8, A0 = 9, RESET = 7 (Freetronics
OLED)
//U8GLIB_SSD1351_128X128GH_HICOLOR u8g(8, 9, 7); // Arduino,
HW SPI Com: SCK = 76, MOSI = 75, CS = 8, A0 = 9, RESET = 7
(Freetronics OLED)
void draw(void) {
   // graphic commands to redraw the complete screen should be placed
here
   u8g.setFont(u8g_font_unifont);
   //u8g.setFont(u8g_font_osb21);
   u8g.drawStr( 0, 22, "Hello World!");
}
void setup(void) {
```

```
    // flip screen, if required
    // u8g.setRot180();
      // set SPI backup if required
    //u8g.setHardwareBackup(u8g_backup_avr_spi);
  // assign default color value
    if ( u8g.getMode() == U8G_MODE_R3G3B2 ) {
        u8g.setColorIndex(255);         // white
    }
    else if ( u8g.getMode() == U8G_MODE_GRAY2BIT ) {
        u8g.setColorIndex(3);                // max intensity
    }
    else if ( u8g.getMode() == U8G_MODE_BW ) {
        u8g.setColorIndex(1);                // pixel on
    }
    else if ( u8g.getMode() == U8G_MODE_HICOLOR ) {
        u8g.setHiColorByRGB(255,255,255);
    }}
 void loop(void) {
    // picture loop
    u8g.firstPage();
    do {
        draw();
```

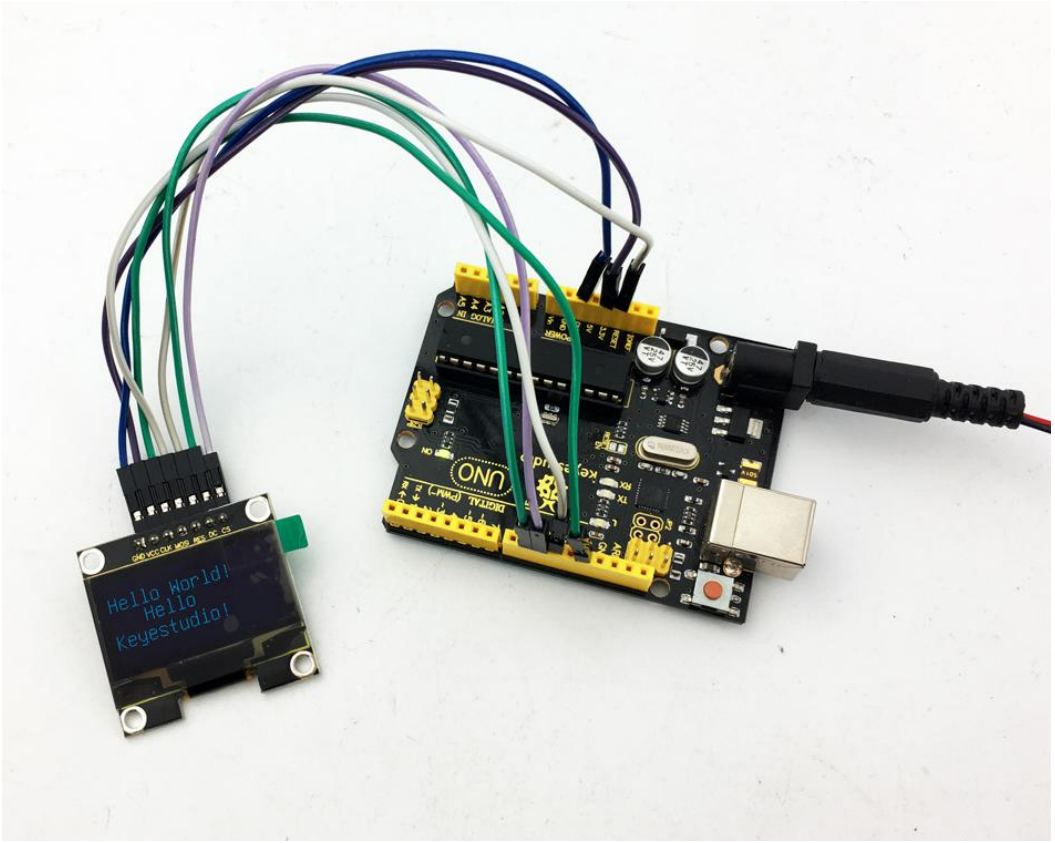
```
    } while( u8g.nextPage() );

      // rebuild the picture after some delay

    delay(50);

}
```

****



Resource

https://fs.keyestudio.com/KS0056