

```

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

// -----
// Hardware-Konfiguration
// -----
#define LS1_PIN 5
#define LS2_PIN 6
#define BUTTON_PIN 7

#define WARTEN 4000    //Wartezeit nach der Messung in ms

// OLED Display
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// Maßstäbe
const float massstaebe[] = {220, 160, 120, 87, 45, 32, 22.5};
const int anzahlMassstaebe = sizeof(massstaebe) /
sizeof(massstaebe[0]);
int aktuellerMassstab = 0;

const double DISTANZ = 100.0; // Abstand der Lichtschranken in
Millimetern;
// Messvariablen
//unsigned long zeitLS1 = 0;
//unsigned long zeitLS2 = 0;
unsigned long startvalue = 0;
unsigned long timediff = 0;

//bool ls1Triggered = false;
//bool ls2Triggered = false;

//int lastLS1State = HIGH;
//int lastLS2State = HIGH;

unsigned long letzteButtonZeit = 0;
const unsigned long debounce = 200;

// -----
// Hilfsfunktionen
// -----
String fmtFloat(float v) {
    if (abs(v - (int)v) < 0.01) return String((int)v);
    return String(v, 2);
}

String fmtFloatm(float v) {
    if (abs(v - (int)v) < 0.01) return String((int)v);
    return String(v, 0);
}

```

```

// -----
// Setup
// -----
void setup() {
    pinMode(LS1_PIN, INPUT_PULLUP);
    pinMode(LS2_PIN, INPUT_PULLUP);
    pinMode(BUTTON_PIN, INPUT_PULLUP);

    display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
    display.clearDisplay();
    display.setTextColor(SSD1306_WHITE);
    display.setTextSize(1);
    display.setCursor(0,0);
    display.display();
}

// -----
// Hauptprogramm
// -----
void loop() {

    // Maßstab wechseln
    if (digitalRead(BUTTON_PIN) == LOW) {
        if (millis() - letzteButtonZeit > debounce) {
            aktuellerMassstab = (aktuellerMassstab + 1) %
anzahlMassstaebe;
            letzteButtonZeit = millis();
        }
    }
    // Statusanzeige
    display.clearDisplay();
    display.setTextSize(2);
    display.setCursor(0,0);
    display.println("Bereit");
    display.setCursor(0,20);
    display.print("1:");
    display.print(fmtFloat(massstaebe[aktuellerMassstab]));
    display.display();

    //Lichtschraken abfragen

    if (digitalRead(LS1_PIN) == LOW) {
        startvalue = millis();
        while (digitalRead(LS2_PIN) == HIGH) {
            wait_display();
        }
        timediff = millis() - startvalue;
        update_display();
    }

    if (digitalRead(LS2_PIN) == LOW) {
        startvalue = millis();
        while (digitalRead(LS1_PIN) == HIGH) {

```

```

        wait_display();
    }
    timediff = millis() - startvalue;
    update_display();
}

} // End of Loop

// Warte auf Schranke
void wait_display(){
    display.clearDisplay();
    display.setTextSize(1);
    display.setCursor(0,0);
    display.println("Warte auf 2. Schranke");
    display.display();
}

// Update Messwert
void update_display(){

    float v_ms = DISTANZ / timediff;
    float v_kmh = v_ms * 3.6;

    float modell_kmh = v_kmh * massstaebe[aktuellerMassstab];
    float modell_ms = modell_kmh / 3.6;

    // Display-Ausgabe
    display.clearDisplay();

    // 1. Zeile: Realwert m/s (groß)
    display.setTextSize(1);
    display.setCursor(0,0);

    display.print(v_ms);
    display.print(" m/s ");
    display.print(timediff);
    display.println(" ms");

    // 2. Zeile: Modell km/h (groß)
    display.setCursor(0,20);
    display.setTextSize(2);
    display.print(modell_kmh);
    display.println(" km/h");

    // 3. Zeile: Modell m/s (klein)

    display.setCursor(0,45);
    display.print(modell_ms);
    display.print(" m/s");

    display.display();

    delay(WARTEN);

```

```
//while (digitalRead(BUTTON_PIN) == HIGH) {} // Diese Zeile  
verwenden, wenn statt der Wartezeit mittels Tastendruck eine neue  
Messung starten soll.  
}
```