Supporting Information 2:
Rotating Disk Electrode for:

Open-Source Equipment Design for Cost-Effective Redox Flow Battery Research

Trinh V. Dung [1], Nguyen T. T. Huyen [1], Nguyen L. T. Huynh [2, 3], Nguyen T. Binh [1], Nguyen T. Dat [1], Nguyen T. T. Nga [1], Nguyen T. Lan [1], Hoang V. Tran [1], Nguyen T. T. Mai [1] and Chinh D. Huynh [1,*]

[1] School of Chemical Chemistry and Life Science, Hanoi University of Science and Technology, Hanoi 100000, Vietnam;
[2] University of Science, Ho Chi Minh City, 700000 Vietnam
[3] Viet Nam National University Ho Chi Minh City, Ho Chi Minh City 700000, Vietnam;
*Correspondence: chinh.huynhdang@hust.edu.vn (C.D.H)

All the printed parts are provided in the following folder. For the experimental setup involving Rotating Disk Electrodes (RDEs) and Opensens potentiostat, a standard divided three-electrode glass cell was employed for cyclic voltammetry experiments. The electrolyte temperature was meticulously maintained at 30°C. In studying the cerium half-cell reactions, RDEs served as the working electrodes, while a platinized titanium mesh was utilized as the counter electrode. The electrode potentials were accurately measured against a saturated Calomel reference electrode, ensuring precise and reliable readings.

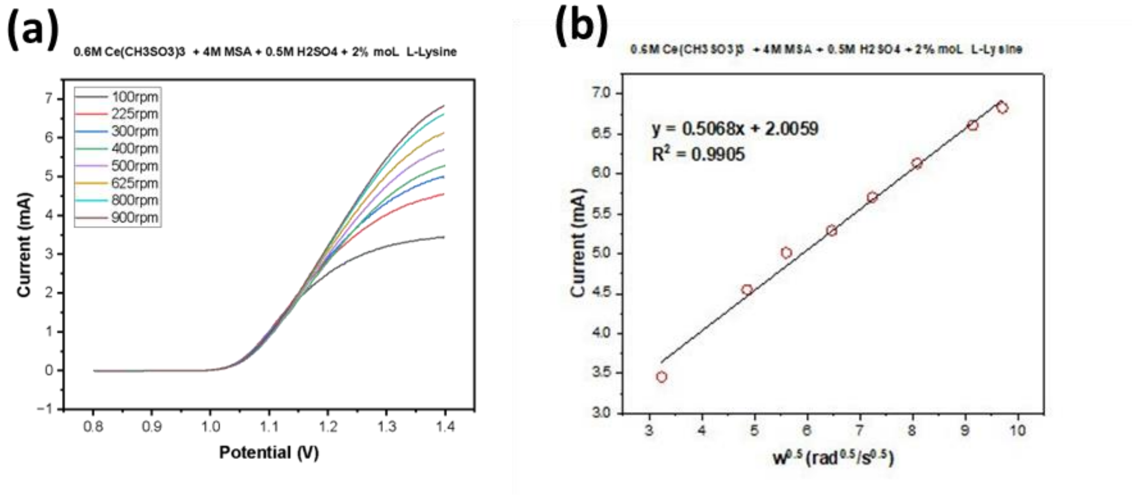

Figure SI 2.1: *(a) Polarization curves in $Ce(CH_3SO_3)_3$ +4M MSA +0.5M H_2SO_4 + 2% mol L-Lysine electrolyte at different rotating speeds. (b) Levich plot of the square root of rotation rate vs the current.*

The electrochemical measurements were conducted using an Opensens system. To analyze the cerium (III/IV) redox reaction, Linear Sweep Voltammetry (LSV) was performed. This involved a linear sweep of the potential from 0.8V to 1.4V versus the Calomel reference, at a scan rate of 10 mV/s. This methodical approach allowed for detailed observation and analysis of the electrochemical behavior of cerium in the specified potential range, providing valuable insights into its redox characteristics under controlled experimental conditions.

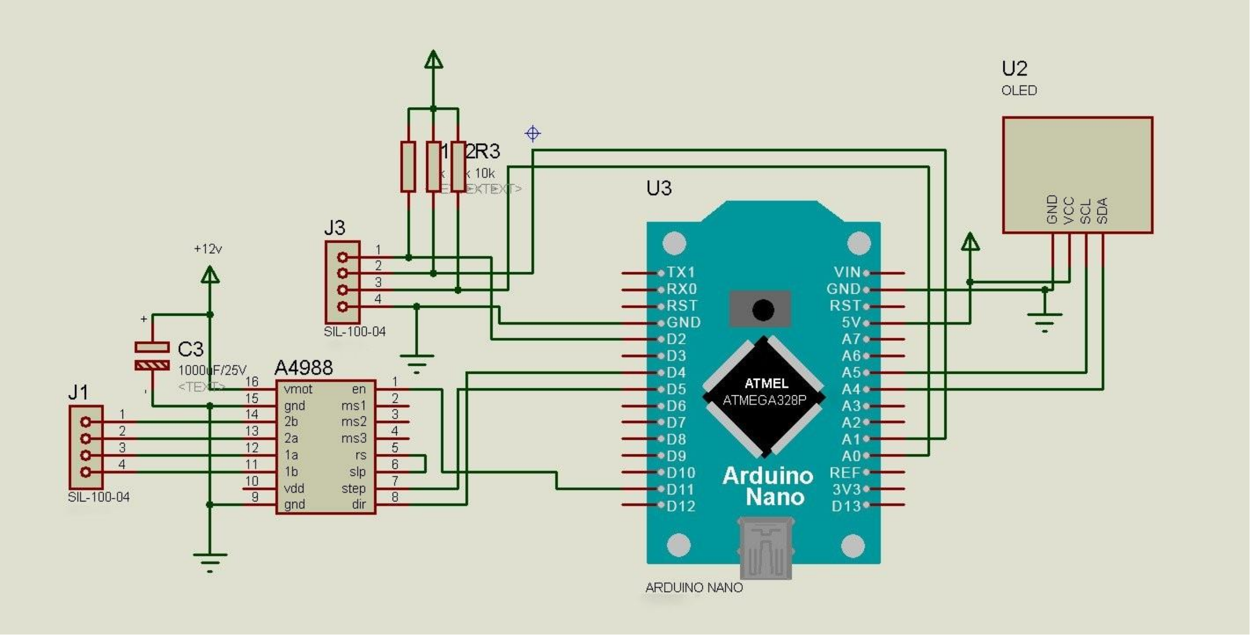

Figure SI 2.2: The schematic of rotating electrode controller

Table SI 2.1: Bill of materials for RDE controller

Device	Part Number	Qty	Unit Price (USD)	Ext. Price (USD)	Description
ARDUINO NANO	1050-1001-ND	01	02.66	02.66	ARDUINO NANO ATMEGA328 EVAL BRD
OLED display	392-CSANAVIDISPLAY1	01	06.50	06.50	Mini 0.96" OLED I2C display
10K	279-YR1B10KCC	03	00.63	01.89	Through Hole Resistors
1000uF 25V	667-EEU-EB1E102B	01	00.90	00.90	Aluminum Electrolytic Capacitors
A4988 module	474-ROB-12859	01	19.95	19.95	Stepper Motor Controller

RDE CODE

```
//oled
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SH1106.h>
#define SCREEN_WIDTH 128  // OLED display width, in pixels
#define SCREEN_HEIGHT 64  // OLED display height, in pixels
#define OLED_RESET 4
Adafruit_SH1106 display(OLED_RESET);
inline void DisplayStuff(uint16_t counter);
inline void Display(uint16_t counter);
// Rotary Encoder Inputs
#include <EEPROM.h>
#include <ClickEncoder.h>
#include <TimerOne.h>
ClickEncoder *encoder;
#define SW 2
```

```
int16_t last, value;
//motor
#define enA 5
//int ledPin = 13;
byte n = 1;
byte u = 0;
int getu = 0;
int go;
int addr = 0;
int counter = 0;
int currentStateCLK;
int lastStateCLK;
String currentDir = "";
unsigned long lastButtonPress = 0;
//stepper

static const unsigned char PROGMEM image_data_mmulogo[] = {
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xFF, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFF, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3F, 0xFE, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7F, 0xFC, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xFF, 0xF0, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xFF, 0xC0, 0x00, 0x03, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFF, 0x00, 0x00, 0x0F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFC, 0x00, 0x00, 0x3F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFE, 0x00, 0x00, 0x7F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFF, 0x00, 0x01, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFF, 0xC0, 0x07, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFF, 0xE0, 0x0F, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFF, 0xF8, 0x3F, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFF, 0xFE, 0xFF, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xCF, 0xFF, 0xFF, 0xE7, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC7, 0xFF, 0xFF, 0x87, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC1, 0xFF, 0xFF, 0x07, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x7F, 0xFE, 0x07, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x3F, 0xF8, 0x07, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x1F, 0xE0, 0x07, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x07, 0xC0, 0x07, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x03, 0x80, 0x07, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x00, 0x00, 0x07, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x00, 0x00, 0x07, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x00, 0x00, 0x0F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x00, 0x00, 0x1F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x00, 0x00, 0x3F, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x00, 0x00, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xC0, 0x00, 0x03, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0x80, 0x00, 0x07, 0xFF, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x00, 0x00, 0x1F, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7F, 0xFC, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xFF, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xFE, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xF8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0F, 0xF0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

void timerIsr() {
  encoder->service();
}

void ngat() {

  int btnState = digitalRead(SW);

  //If we detect LOW signal, button is pressed
  if (btnState == LOW) {
    //if 50ms have passed since last LOW pulse, it means that the
    //button has been pressed, released and pressed again
    if (millis() - lastButtonPress > 50) {
      // Serial.println("Button pressed!");

      if (n == 0) {
        n = 1;
        u = 0;
      } else {
        n = 0;
      }
      //Display(counter);

      //encoder();
    }

    // Remember last button press event
    lastButtonPress = millis();
  }
```

```
}

void encoder_rorate() {

  while (n == 1) {
    digitalWrite(5, HIGH);
    if (u == 0) {
      DisplayStuff(counter);
    }
    u = 1;
    counter += encoder->getValue();
    if (counter <= 400) {
      counter = 400;
    }
    if (counter >= 2500) {
      counter = 2500;
    }
    if (counter != last) {
      last = counter;
      Serial.print("Encoder counter: ");
      Serial.println(counter);
    }
    DisplayStuff(counter);
    int val = counter / 12;
    EEPROM.update(0, val);
    addr = addr + 1;
    if (addr == EEPROM.length())
      addr = 0;
    delay(5);  // delay 5ms để trước khi lưu giá trị tiếp theo
  }
  return counter;
}

void setup() {
  // Setup Serial Monitor
  Serial.begin(9600);
  //setup encoder
  encoder = new ClickEncoder(A1, A0, 2);
  Timer1.initialize(1000);
  Timer1.attachInterrupt(timerIsr);
  last = -1;
  //oled
  display.begin(SH1106_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();                                    //clear display buffer
  display.drawBitmap(0, 0, image_data_mmulogo, 128, 64, 1);  //draw boot screen
  display.setTextSize(1.5);                                  //text size small
  display.setTextColor(WHITE);                               //white text - might not be needed
  display.setCursor(30, 50);                                 //from 35X 10Y
  display.println("opensens.io");                            //print text
  display.display();                                         //display boot screen
  delay(2000);
  //delay(20000);
  //stepper
  pinMode(11, OUTPUT);
  digitalWrite(11, LOW);
  digitalWrite(5, HIGH);
  // Set encoder pins as inputs
  pinMode(SW, INPUT_PULLUP);
  pinMode(2, INPUT_PULLUP);
  //pinMode(ledPin, OUTPUT);
  attachInterrupt(0, ngat, LOW);
  getu = EEPROM.read(addr);
  delay(5);
  counter = getu * 12;  // Read the current state of CLK
  //counter = 1;
  DisplayStuff(counter);
}

void loop() {

  encoder_rorate();
  if (u == 1) {
    Display(counter);
    u = 0;
  }
  go = counter * 0.1;
  int pwmOutput = map(go, 1023, 0, 0, 1023);  // Map the potentiometer value from 0 to 255
  analogWrite(enA, pwmOutput);                    // Send PWM signal to L298N Enable pin
}



inline void DisplayStuff(uint16_t counter) {
  display.clearDisplay();       //clear display buffer
  display.setTextSize(2);       //text size small
  display.setTextColor(WHITE);  //white text - might not be needed
  display.setCursor(22, 15);    //from 35X 10Y
  display.println("SETTING");   //print text
  display.setTextSize(3);       //text size medium
  display.setCursor(30, 40);    //from 15X 30Y
  display.print(counter);       //print rpm value
  display.setTextSize(1);       //text size small
  display.println(" RPM");      //print text after previous, same line
  display.display();            //display buffer
}
inline void Display(uint16_t counter) {
  display.clearDisplay();         //clear display buffer
  display.setTextSize(2);         //text size small
  display.setTextColor(WHITE);    //white text - might not be needed
  display.setCursor(5, 10);       //from 35X 10Y
  display.println("RDE SYSTEM");  //print text
  display.setTextSize(3);         //text size medium
  display.setCursor(30, 40);      //from 15X 30Y
  display.print(counter);         //print rpm value
  display.setTextSize(1);         //text size small
  display.println(" RPM");        //print text after previous, same line
  display.display();              //display buffer
}
```