ДОДАТОК А

(Обов’язковий)

КОД ПРОГРАМИ

Файл: Suslovets

```
#include <WiFi.h>
#include <HTTPClient.h>
const char* ssid     = "suslovets-ssid";
const char* password = "suslovets-password";
const char* server = "www.bike-suslovets.com/update";
HTTPClient http;

#define GERCON_PIN 2
#define DIAMETER 26
#define CONVERT_MS_TO_KMH 3.6
#define DELATA_DISTANCE DIAMETER * 3.14 * 2.54
#define MIN_ITERVAL 2000

float kmh = 0.0;
float distance = 0.0;
float deltaD = 0.0;
boolean move = false;
unsigned long first = 0;
unsigned long count = 0;
unsigned long deltaTime = 0;

float current = 0;
float voltage = 0;
float power = 0;

#define ADC 10
#define ADC_CAPACITY 4094
#define VOLTAGE_BAT 48
#define COEFFICIENT (VOLTAGE_BAT/VOLTAGE_REFERENCE)

#define CURRENT_SENSE_PIN 3
#define VOLTAGE_REFERENCE 3.3
#define SENSITIVITY 0.04 // 40mV/A -Sensitivity of ACS758 (check the datasheet
for your ACS758 model)

const float OFFSET_VOLT = VOLTAGE_REFERENCE / 2; // Zero-current output voltage (
Offset Voltage)

#include "RTClib.h"
RTC_DS3231 rtc;
char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday",
"Thursday", "Friday", "Saturday"};

#include "SD_MMC.h"
#define FILE_NAME "/BikeBlakBox.txt"

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

```
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
#define I2C_ADDRES_OLED 0x3C
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

void setup() {
  pinMode(GERCON_PIN, INPUT);         //setting gercon pin to read
  pinMode(CURRENT_SENSE_PIN, INPUT);  //Setting current pin to read

  Serial.begin(9600);

  rtc.begin();    // init rtc module
  rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));     // set date and time

  initSD();       //init SD module
  createdFile(FILE_NAME);  //created file for record

  if(!display.begin(SSD1306_SWITCHCAPVCC, I2C_ADDRES_OLED)) {    // Address 0x3D
for 128x64
    Serial.println(F("SSD1306 failed !"));
  } else {
    oledInit();     //display first screen
  }

  Serial.println("\n[WiFi] Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password); // init wifi

  http.begin(server); //init http
  // Will try for about 10 seconds (20x 500ms)
  int tryDelay = 500;
  int numberOfTries = 20;
  // Wait for the WiFi event
  while (true) {
    switch(WiFi.status()) {
      case WL_NO_SSID_AVAIL:
        Serial.println("[WiFi] SSID not found");
        break;
      case WL_CONNECT_FAILED:
        Serial.print("[WiFi] Failed - WiFi not connected! Reason: ");
        return;
        break;
      case WL_CONNECTION_LOST:
        Serial.println("[WiFi] Connection was lost");
        break;
      case WL_SCAN_COMPLETED:
        Serial.println("[WiFi] Scan is completed");
        break;
      case WL_DISCONNECTED:
```

```
      Serial.println("[WiFi] WiFi is disconnected");
      break;
    case WL_CONNECTED:
      Serial.println("[WiFi] WiFi is connected!");
      Serial.print("[WiFi] IP address: ");
      Serial.println(WiFi.localIP());
      return;
      break;
    default:
      Serial.print("[WiFi] WiFi Status: ");
      Serial.println(WiFi.status());
      break;
  }
    delay(tryDelay);

    if(numberOfTries <= 0){
      Serial.print("[WiFi] Failed to connect to WiFi!");
      // Use disconnect function to force stop trying to connect
      WiFi.disconnect();
      return;
    } else {
      numberOfTries--;
    }
}

xTaskCreatePinnedToCore(
                        Task_GERCON,
                        "Task_GERCON",
                        4096,
                        NULL,
                        1,
                        NULL,
                        tskNO_AFFINITY
                        );

xTaskCreatePinnedToCore(
                        Task_POWER,
                        "Task_POWER",
                        4096,
                        NULL,
                        1,
                        NULL,
                        tskNO_AFFINITY
                        );

xTaskCreatePinnedToCore(
                        Task_OLED,
                        "Task_OLED",
                        4096,
                        NULL,
```

```
                          1,
                          NULL,
                          tskNO_AFFINITY
                          );

  xTaskCreatePinnedToCore(
                          Task_SD,
                          "Task_SD",
                          4096,
                          NULL,
                          1,
                          NULL,
                          tskNO_AFFINITY
                          );
}

void Task_GERCON(void *pvParameters) {
  (void)pvParameters;

  while(1){

    if(digitalRead(GERCON_PIN) == HIGH) {
      first = millis();
      move = true;
    }

    if(move && digitalRead(GERCON_PIN) == HIGH && (millis() - first) <
MIN_ITERVAL) {
      deltaTime = millis() - first;
      first = millis();
      count++;
    } else {
      move = false;
      kmh = 0;
    }

    kmh = (1 / deltaTime) * DELATA_DISTANCE * CONVERT_MS_TO_KMH;
    Serial.println("KM/H: " + String(kmh));

    distance = (count * DELATA_DISTANCE) / 1000;    // (count *
DELATA_DISTANCE)in m / 1000 = distance to km
    Serial.println("Distance total: " + String(distance));

    vTaskDelay(10);
  }
}

void Task_POWER(void *pvParameters) {
  (void)pvParameters;
```

```
  while(1){
    current = getCurrent();
    voltage = getVoltage();
    power = getPower(voltage, current);
    vTaskDelay(10);
  }
}

void Task_OLED(void *pvParameters) {
  (void)pvParameters;

  while(1){
    oled();
    vTaskDelay(10);
  }
}

void Task_SD(void *pvParameters) {
  (void)pvParameters;

  while(1){
    recordSD(dataFroSD(), FILE_NAME);
    vTaskDelay(10);
  }
}
void loop() {
  if(WiFi.status() == WL_CONNECTED) {
    readSD(FILE_NAME);  //read data from SD and send to server
  }
}
```

Файл: current_sensor

```
float getCurrent() {
  int raw_adc = analogRead(CURRENT_SENSE_PIN);
  float voltage = raw_adc * (VOLTAGE_REFERENCE / 4095.0);
  float current = (voltage - OFFSET_VOLT) / SENSITIVITY;

  Serial.print("Current: ");
  Serial.print(current);
  Serial.println(" A");

  return current;
}

float getVoltage() {
  float raw = analogRead(ADC);
  float voltage = VOLTAGE_REFERENCE /ADC_CAPACITY * raw * COEFFICIENT;
  return voltage;
```

```
}

float getPower(float voltage, float current) {
  return voltage * current;
}
```

Файл: ds3231

```
String getDate(){
  DateTime now = rtc.now();

    Serial.print(now.year(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.day(), DEC);
    Serial.print(" (");
    Serial.print(daysOfTheWeek[now.dayOfTheWeek()]);
    Serial.print(") ");
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.println();

    return String(now.TIMESTAMP_FULL);
}
```

Файл: oled

```
void oledInit() {
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,20);
  display.println("Suslovets Ruslan");
  display.setCursor(20,20);
  display.println("TRRTy-21-1");
  display.setCursor(40,60);
  display.println("2024");
  display.display();
  delay(2000);
}

void oled() {
  display.clearDisplay();
```

```
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,20);
  display.println("km/h: ");
  display.println(kmh);
  display.setCursor(20,20);
  display.println("distance km: ");
  display.println(distance);
  display.setCursor(40,60);
  display.println("voltage bat: ");
  display.println(voltage);
  display.setCursor(60,60);
  display.println("current: ");
  display.println(current);
  display.display();
}
```

Файл: sd

```
void initSD() {
  if(!SD_MMC.begin()){
    Serial.println("Failed to mount card");
    return;
  }
}

void createdFile(String fileName) {
  File file = SD_MMC.open(fileName, FILE_WRITE);

  if(!file){
    Serial.println("Opening file failed");
    return;
  }

  if(file.print("Started record:\n")){
        Serial.println("File write success");
    } else {
        Serial.println("File write failed");
    }
    file.close();
}

void recordSD(String data, String fileName) {
  File file = SD_MMC.open(fileName, FILE_WRITE);

  if(file) {
    if(file.println(data)) {
      Serial.println("Data write to SD succeess");
    } else {
```

```
      Serial.println("Data write failed !");
    }
  } else {
    Serial.println("Opening file failed !");
  }
}

void readSD(String fileName) {
  File file = SD_MMC.open(fileName, FILE_READ);

  if(file) {
    while(file.available()) {
      Serial.println(file.read());
      sendToServer(String(file.read()));
    }
  } else {
    Serial.println("Opening file failed !");
  }
}

String dataFroSD() {
  return String(getDate()
    + " : km/h - " + kmh
    + " distance - " + distance
    + " voltage - " + voltage
    + " current - " + current
    + " power - " + power);
}
```

Файл: server

```
void sendToServer(String data) {
  if(http.GET() == 200) {
    http.POST(data);
  } else {
    Serial.println("Server not found!");
  }
   http.end();
}
```

ДОДАТОК Б

(Рекомендований)

КОПІЇ ПРЕЗЕНТАЦІЇ

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки
Кафедра Радіотехнологій інформаційно-комунікаційних систем

Графічні матеріали до кваліфікаційної роботи бакалавра

на тему:

ПРИСТРІЙ ЗБОРУ СТАТИСТИКИ ПАРАМЕТРІВ ЕЛЕКТРОВЕЛОСИПЕДА

Виконав: студент 3 курсу, групи ТРРТу-21-1
Освітня програма «Радіотехніка»
Сусловець Руслан Ігорович
Керівник: ст. викладач Ганшин Дмитро Г.

Актуальність

2

Історія винаходу велосипеда

Крікпатрік Макміллан разом зі своїм прототипом велосипеда

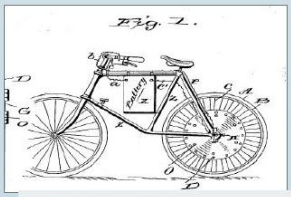
1839 рік

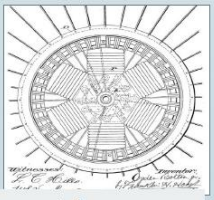
1903 рік

1895 рік

2000-ті роки

Перший у світі переможець «Тур де Франс» - Моріс гарен





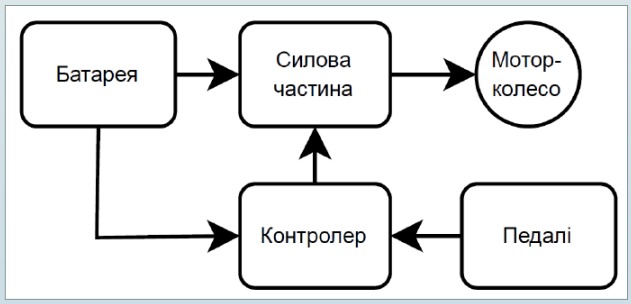
Приблизний вигляд першого в світі запатентованого велосипеду з електродвигуном

Один з варіантів «conversion kit»

3

Основні компоненти електровелосипеда

- **Контролер**. Керує перемиканням на основі робочого циклу, відповідно до вимог користувача та напруги акумулятора;
- **Силова частина**. Являє собою перетворювач живлення, що з'єднує акумулятор з двигуном, регулюючи подачу струму;
- **Батарея**. На даний момент найбільш популярними є низьковольтні літій-іонні акумулятори;
- **Електродвигун**. Перетворює електричну енергію у механічну.



4

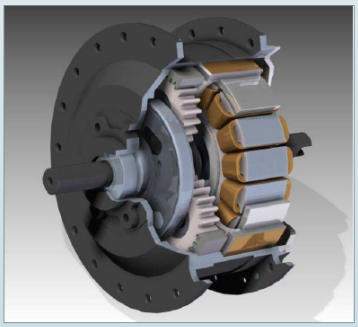
Вибір типу електродвигуна електровелосипеда

Редуктор - являє собою механічний пристрій, котрий використовується для збільшення крутного моменту між входом та на виході з нього

Редукторні мотор-колеса використовуються у електросамокатах та багато у яких видах електротранспорту.

Основні переваги

- Підвищений крутний момент;
- Малі розміри та вага;
- Більша економнічність;
- Доступна ціна.



Внутрішня будова редукторного колеса

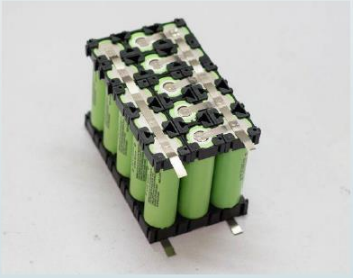
5

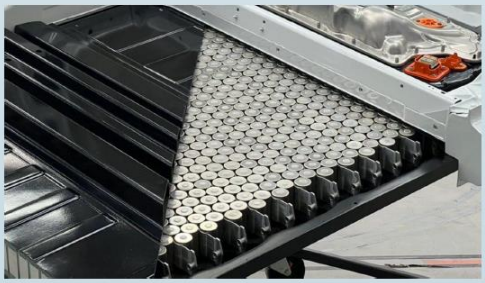
Вибір типу акумулятора

Основний тип акумулятора для електровелосипеда - літій-іонна батарея, об'єднана у блоки з акумуляторів типу 18650.

Переваги:

- Висока швидкість заряду;
- Відносно гнучка форма кінцевої батареї;
- Гнучкість у характеристиках.





Яскравий та відомий приклад – акумуляторні батареї в автомобілях марки Tesla.

Відома та перевірена часом технологія сприяє підвищенню запасу надійності кінцевого виробу.

6

Огляд контролерів що використовуються

	Arduino	STM32	ESP32
Вбудовані модулі	-	Wi-Fi, Bluetooth v4.2	Wi-Fi і Bluetooth
Процесор	ATmega328P	ARM-ядро CORTEX-M3	Двоядерний процесор Tensilica Xtensa
Тактова чатота процесора	16 МГц	72 МГц	До 240 Мгц
Підтримувальні інтерфейси	USB, UART, SPI та I2C.	SD, UART, SPI, SDIO, I²C, LED PWM, Motor PWM, I²S, I²C, IR GPIO, сенсорний датчик, ADC, DAC, LNA.	Wi-Fi, Bluetooth, I2C, SPI, UART, PWM, ADC і DAC, 10 портів для сенсорних датчиків, 34 виходи GPIO, інфрачервоний контролер.

Перевага мікроконтролера на платформі ESP32 над іншими - це простота у використанні і, водночас, висока продуктивність.

7

Апаратна частина пристрою

Основні показники:

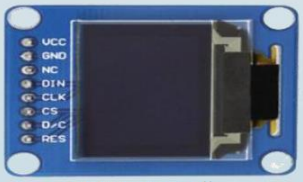
1. Швидкість руху електровелосипеду ;
2. Споживання струму в А;
3. Напруга батареї в В;
4. Потужність споживання енергії.

Структурна схема пристрою

Геркон

Модуль ACS758LCB

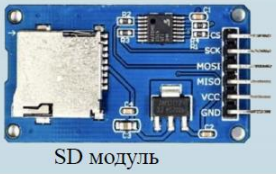
ESP32



OLED дисплей

Модуль реального часу DS3231

Сервер



SD модуль

8

Програмна частина пристрою

FreeRTOS — багатозадачна операційна система реального часу для вбудованих систем.

Ключові переваги та особливості FreeRTOS:

- Керування задачами
- Планувальник
- Міжзадачне спілкування
- Таймери та затримки
- Малий розмір коду
- Управління пам’яттю

9

Програмна частина пристрою

Структура всієї програми – це шість файлів з описом функціоналу:

current_sensor	функції для роботи з модулем ACS758LCB
ds3231	функції для роботи з модулем реального часу DS3231
oled	функції для роботи з oled дисплеєм
sd	функції для роботи з SD модулем
server	функції для відправки даних з флеш карти на сервер при підключені до Wi-Fi мережі
Suslovets	основний код програми

Кожен з них виконує свою роботу:

- Зчитування показників споживання енергії;
- Визначення поточного часу;
- Вивід інформації на екран;
- Запис основних даних на SD карту та передавання їх на сервер.

10

Програмна частина пристрою

Після оголошення змінних та використовуваних бібліотек описуємо функцію підключення до Wi-Fi мережі та можливі помилки, на випадок якщо такі виникнуть.

```
while (true) {
  switch(WiFi.status()) {
    case WL_NO_SSID_AVAIL:
      Serial.println("[WiFi] SSID not found");
      break;
    case WL_CONNECT_FAILED:
      Serial.print("[WiFi] Failed - WiFi not connected! Reason: ");
      return;
      break;
    case WL_CONNECTION_LOST:
      Serial.println("[WiFi] Connection was lost");
      break;
    case WL_SCAN_COMPLETED:
      Serial.println("[WiFi] Scan is completed");
      break;
    case WL_DISCONNECTED:
    Serial.println("[WiFi] WiFi is disconnected");
    break;
  case WL_CONNECTED:
    Serial.println("[WiFi] WiFi is connected!");
    Serial.print("[WiFi] IP address: ");
    Serial.println(WiFi.localIP());
    return;
    break;
  default:
    Serial.print("[WiFi] WiFi Status: ");
    Serial.println(WiFi.status());
    break;
}
```

11

Програмна частина пристрою

Наступний крок – підв'язка файлів у основний код програми.

Кожен з «тасків» запускає відповідну частину коду, з описом, потрібних у цей проміжок часу, функцій:

«Task_Gercon» - розрахунок і зчитування кількості обертів колеса;

«Task_Power» - зняття даних з датчику струму і розрахунок споживання;

«Task_Oled» - виведення отриманої інформації на екран;

«Task_Sd» - запис цієї ж самої інформації на флеш-карту для подальшої передачі на сервер.

```
xTaskCreatePinnedToCore(
                        Task_GERCON,
                        "Task_GERCON",
                        4096,
                        NULL,
                        1,
                        NULL,
                        tskNO_AFFINITY
                        );

xTaskCreatePinnedToCore(
                        Task_POWER,
                        "Task_POWER",
                        4096,
                        NULL,
                        1,
                        NULL,
                        tskNO_AFFINITY
                        );

xTaskCreatePinnedToCore(
                        Task_OLED,
                        "Task_OLED",
                        4096,
                        NULL,
                        1,
                        NULL,
                        tskNO_AFFINITY
                        );

xTaskCreatePinnedToCore(
                        Task_SD,
                        "Task_SD",
                        4096,
                        NULL,
                        1,
                        NULL,
                        tskNO_AFFINITY
```

Висновок

Проаналізовано основні компоненти електровелосипеду та виділено наступне:

- батарея, яка складається з літій-іонних акумуляторів;
- силова частина, тобто контроль живлення електродвигуна;
- електродвигун.

Під час виконання кваліфікаційної роботи були застосовані теоретичні знання та створили пристрій збору статистики основних параметрів електровелосипеду:

- швидкість;
- дистанція;
- напруга акумуляторної батареї;
- струм споживання електровелосипеду;
- загальна потужність споживання електровелосипеду.

Зібрані дані зберігаються на флеш карту з в казанням часу та дати запису параметрів.

Розроблено автоматичне відправлення записаних даних до серверу для подальшого зберігання даних та обробки.

13

ДОДАТОК В

(Обов’язковий)

№ п/п	Позначення	Найменування	Додаткові відомості
		Текстові документи	
1		Пояснювальна записка	76 с.

Зм	Арк.	№ докум.	Підп.	Дата	ГЮІК.ХХХХХХ.000Д2			
Розроб.		Сусловець			Пристрій збору статистики параметрів електровелосипеда.	Літера	Аркуш	Аркушів
Перевір.		Бітченко				У	1	1
Н. контр.		Зарудний			Відомість кваліфікаційної роботи.	ХНУРЕ Кафедра РТІКС		
Затв.		Цопа						