ДОДАТОК А

Код передавача

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SPI.h>
#include <RF24.h>

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

RF24 radio(9, 10); // CE, CSN
const byte address[6] = "00001";

const int pinJoyX = A0;
const int pinJoyY = A1;
const int btnA = 2;
const int btnC = 4;
const int btnB = 3;
const int btnD = 5;

int menuIndex = 0;
const int menuCount = 4;
bool inSubmenu = false;
int submenuIndex = -1;

bool calibrating = false;
int joyMinX = 1023, joyMaxX = 0;
int joyMinY = 1023, joyMaxY = 0;
bool isCalibrated = false;

struct JoystickData {
  int x;
  int y;
};

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

  pinMode(btnA, INPUT_PULLUP);
  pinMode(btnB, INPUT_PULLUP);
  pinMode(btnC, INPUT_PULLUP);
```

```
  pinMode(btnD, INPUT_PULLUP);

  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println("OLED init failed");
    while (1);
  }

  // nRF24 setup
  radio.begin();
  radio.setPALevel(RF24_PA_LOW);
  radio.setDataRate(RF24_1MBPS);
  radio.setChannel(100);
  radio.openWritingPipe(address);
  radio.stopListening();

  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.display();
}

void loop() {
  if (calibrating) {
    calibrateJoystickLive();
  } else if (!inSubmenu) {
    showMenu();
    handleMenuInput();
  } else {
    if (submenuIndex == 0) {
      showJoystickView();
    } else if (submenuIndex == 1) {
      showInfoView();
    } else if (submenuIndex == 2) {
      startCalibration();
    } else if (submenuIndex == 3) {
      showVersionView();
    }
  }
  delay(100);
}

void sendJoystickData(int x, int y) {
```

```
  JoystickData data = {x, y};
  radio.write(&data, sizeof(data));
  Serial.print("Sent: X=");
  Serial.print(x);
  Serial.print(" Y=");
  Serial.println(y);
}

void showMenu() {
  display.clearDisplay();
  display.setTextSize(2);
  display.setCursor((SCREEN_WIDTH - 6 * 5) / 2, 0);
  display.println("Menu");
  display.setTextSize(1);

  for (int i = 0; i < menuCount; i++) {
    display.setCursor(10, 20 + i * 10);
    if (i == menuIndex) display.print("-> ");
    else display.print("   ");

    switch (i) {
      case 0: display.println("Joystick"); break;
      case 1: display.println("Info"); break;
      case 2: display.println("Calibrate"); break;
      case 3: display.println("Version"); break;
    }
  }
  display.display();
}

void handleMenuInput() {
  static bool lastA = HIGH, lastC = HIGH, lastB = HIGH;
  bool a = digitalRead(btnA);
  bool c = digitalRead(btnC);
  bool b = digitalRead(btnB);

  if (a == LOW && lastA == HIGH) {
    menuIndex = (menuIndex - 1 + menuCount) % menuCount;
  }
  if (c == LOW && lastC == HIGH) {
    menuIndex = (menuIndex + 1) % menuCount;
  }
```

```
  if (b == LOW && lastB == HIGH) {
    if (menuIndex == 2) startCalibration();
    else {
      animateTransition();
      inSubmenu = true;
      submenuIndex = menuIndex;
    }
  }

  lastA = a; lastC = c; lastB = b;
}

void animateTransition() {
  display.clearDisplay();
  display.setCursor(40, 25);
  display.setTextSize(1);
  display.println("Loading");
  display.display();

  for (int i = 0; i < 3; i++) {
    display.setCursor(88 + i * 4, 25);
    display.print(".");
    display.display();
    delay(250);
  }

  delay(150);
  display.clearDisplay();
  display.display();
}

void showJoystickView() {
  int x = analogRead(pinJoyX);
  int y = analogRead(pinJoyY);
  int normX = isCalibrated ? map(x, joyMinX, joyMaxX, 0, 100) : map(x, 0, 1023, 0, 100);
  int normY = isCalibrated ? map(y, joyMinY, joyMaxY, 100, 0) : map(y, 0, 1023, 100, 0);

  normX = constrain(normX, 0, 100);
  normY = constrain(normY, 0, 100);

  display.clearDisplay();
  display.setCursor(0, 0);
```

```
  display.println("Joystick View");

  int boxX = 70, boxY = 20, boxW = 50, boxH = 40;
  display.drawRect(boxX, boxY, boxW, boxH, SSD1306_WHITE);
  int px = boxX + map(normX, 0, 100, 0, boxW - 1);
  int py = boxY + map(normY, 0, 100, 0, boxH - 1);
  display.fillCircle(px, py, 2, SSD1306_WHITE);

  display.setCursor(0, 48);
  display.print("X: "); display.print(normX);
  display.print(" Y: "); display.print(normY);
  display.display();

  sendJoystickData(normX, normY);

  if (digitalRead(btnD) == LOW) {
    inSubmenu = false;
    submenuIndex = -1;
    delay(300);
  }
}

void showInfoView() {
  display.clearDisplay();
  display.setCursor(0, 0);
  display.println("Info");
  display.println("Joystick Menu Demo");
  display.println("A/C - Navigate");
  display.println("B - Select");
  display.println("D - Back");
  display.display();

  if (digitalRead(btnD) == LOW) {
    inSubmenu = false;
    submenuIndex = -1;
    delay(300);
  }
}

void showVersionView() {
  display.clearDisplay();
  display.setCursor(0, 0);
```

```
  display.println("Version");
  display.println("v1.2.3");
  display.println("+ nRF24L01 TX");
  display.display();

  if (digitalRead(btnD) == LOW) {
    inSubmenu = false;
    submenuIndex = -1;
    delay(300);
  }
}

void startCalibration() {
  calibrating = true;
  joyMinX = 1023;
  joyMaxX = 0;
  joyMinY = 1023;
  joyMaxY = 0;
}

void calibrateJoystickLive() {
  int x = analogRead(pinJoyX);
  int y = analogRead(pinJoyY);

  if (x < joyMinX) joyMinX = x;
  if (x > joyMaxX) joyMaxX = x;
  if (y < joyMinY) joyMinY = y;
  if (y > joyMaxY) joyMaxY = y;

  display.clearDisplay();
  display.setCursor(0, 0);
  display.println("Calibrating...");
  display.println("Move stick around");
  display.println("Press B to save");

  display.setCursor(0, 40);
  display.print("X: "); display.print(x);
  display.print(" Y: "); display.print(y);
  display.display();

  if (digitalRead(btnB) == LOW) {
    calibrating = false;
```

```
    isCalibrated = true;
    delay(500);
  }
}
```

ДОДАТОК Б

Код приймача

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(9, 10);
const byte address[6] = "00001";

struct JoystickData {
  int x;
  int y;
};

const int ledForward = 2;
const int ledBackward = 3;
const int ledLeft = 4;
const int ledRight = 5;

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

  pinMode(ledForward, OUTPUT);
  pinMode(ledBackward, OUTPUT);
  pinMode(ledLeft, OUTPUT);
  pinMode(ledRight, OUTPUT);

  if (!radio.begin()) {
    Serial.println("NRF24L01 not found!");
    while (1);
  }

  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_LOW);
  radio.setDataRate(RF24_1MBPS);
  radio.setChannel(100);
  radio.startListening();

  Serial.println("Receiver started and listening...");
}

void loop() {
  if (radio.available()) {
    JoystickData data;
```

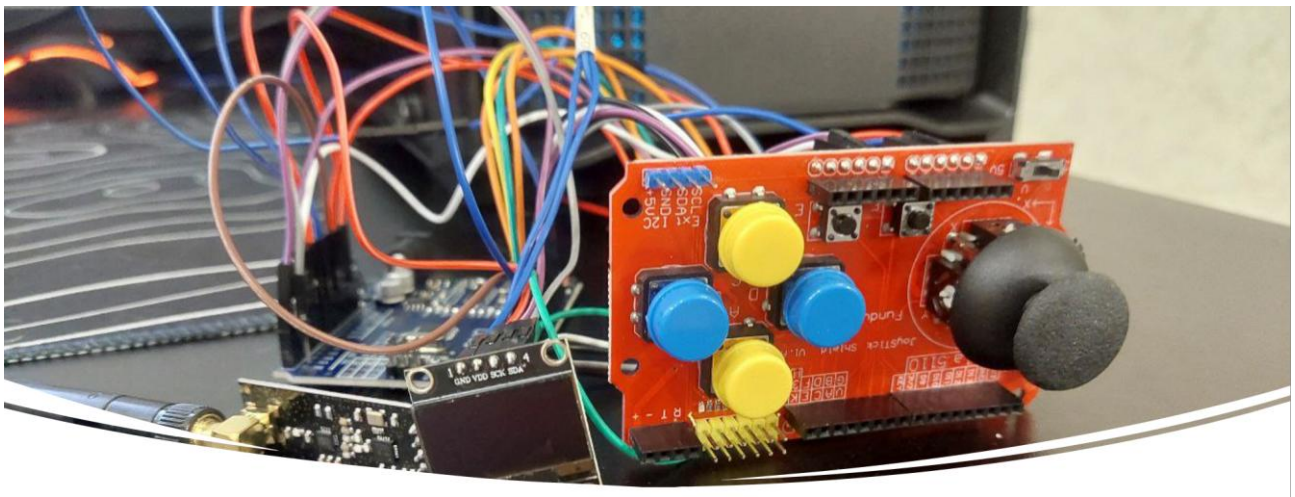
```
    radio.read(&data, sizeof(data));

    Serial.print("Received: X=");
    Serial.print(data.x);
    Serial.print(" Y=");
    Serial.println(data.y);

    // Сброс всех светодиодов
    digitalWrite(ledForward, LOW);
    digitalWrite(ledBackward, LOW);
    digitalWrite(ledLeft, LOW);
    digitalWrite(ledRight, LOW);

    const int center = 50;
    const int threshold = 15;

    int deltaX = data.x - center;
    int deltaY = data.y - center;

    if (abs(deltaX) < threshold && abs(deltaY) < threshold) {
      return;
    }

    if (abs(deltaY) >= abs(deltaX)) {
      // По вертикали
      if (deltaY > 0) {
        digitalWrite(ledForward, HIGH);
      } else {
        digitalWrite(ledBackward, HIGH);
      }
    } else {
      // По горизонтали
      if (deltaX > 0) {
        digitalWrite(ledRight, HIGH);
      } else {
        digitalWrite(ledLeft, HIGH);
      }
    }
  }
}
```

ДОДАТОК В
Демонстраційний матеріал

Розроблення пульта дистанційного керування роботом-промоутером

Сильцов Микита Костянтинович
ХНУРЕ, 2025

Навіщо потрібен пульт дистанційного керування?



Ідея проєкту

- Ідея проєкту полягає у створенні пульта дистанційного керування з двостороннім бездротовим зв’язком за допомогою модуля nRF24L01. Інтуїтивний інтерфейс із OLED-дисплеєм і джойстиком забезпечує просте та надійне керування роботом.

Склад пульта

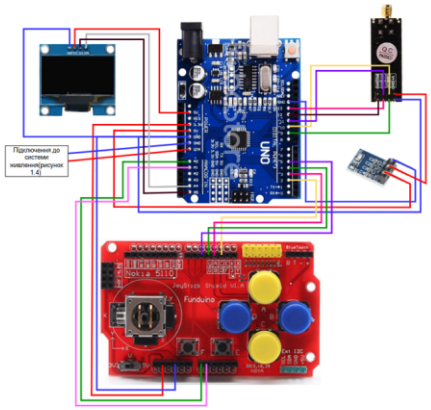


Схема пульту який буде керувати роботом-промоутером

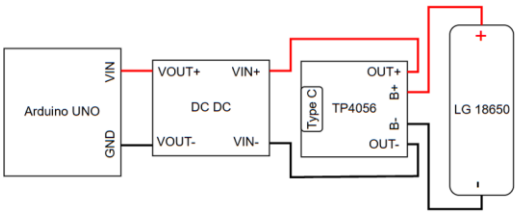


Схема живлення пульта дистанційного керування

Меню на дисплеї

Зовнішній вигляд меню

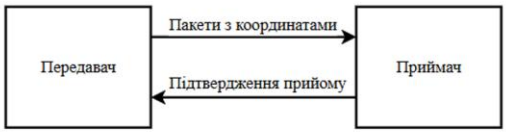
Вкладка Joystick

Вкладка Calibrate

Вкладка Info

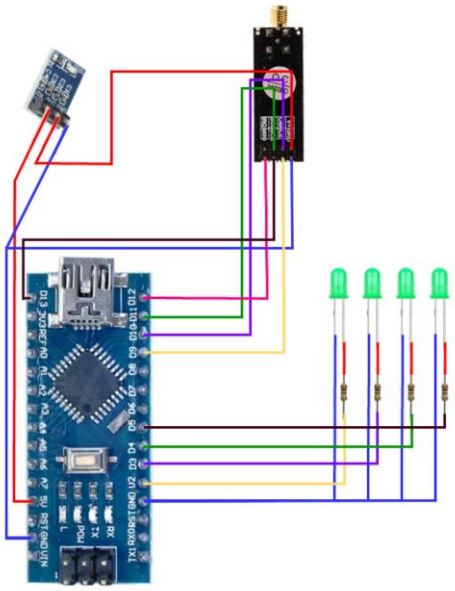
Вкладка Version

Принцип передачі

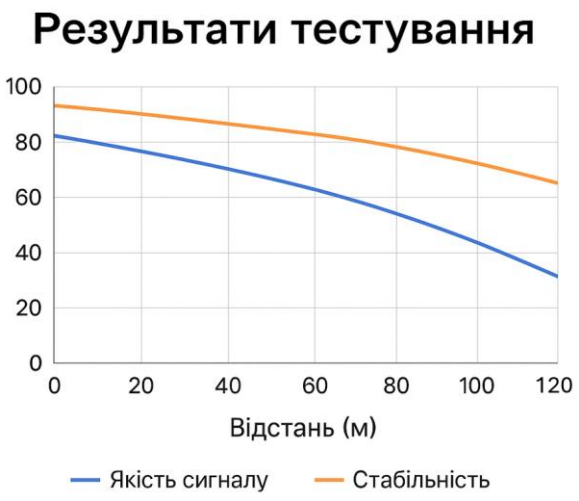


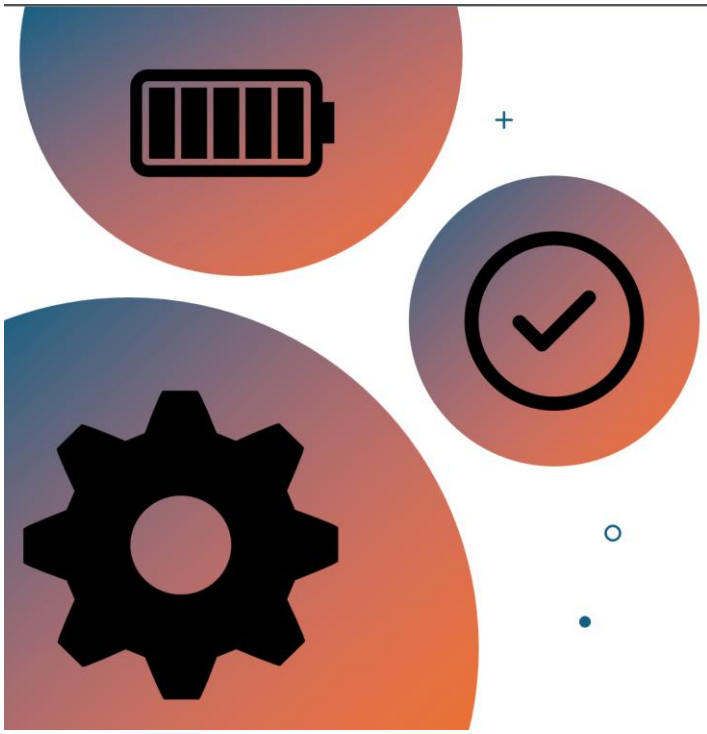
Дані передаються у вигляді структурованих пакетів

Приймач



Результати тестування





Переваги системи

- **Автономність**
  система працює незалежно, не потребує постійного підключення до мережі або додаткових пристроїв
- **Гнучкість**
  можливість налаштувань під різні завдання, адаптація під різні сценарії
- **Простота реалізації**
  легкість в розробці, монтажі та експлуатації

Демонстрація роботи пульта дистанційного керування

Дякую за увагу!

NURE

Харківський національний університет радіоелектроніки

Формат	Позиція.	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кільк.	Примітки
			Текстові документи		
А4	1	ГЮИК 416616.009 ПЗ	Пояснювальна записка	58 с.	
			Інші документи	11 с.	
А4	2		Код передавача	8 с.	
А4	3		Код приймача	3 с.	

Змін.	Арк.	Номер докум.	Підпис	Дата	ГЮИК 416616.009 ВД			
Розроб.		Сильцов М.К.			Розроблення пульту дистанційного керування роботом-промоутером	Літера	Арку	Аркуш
Перевір.		Новоселов С.П.				Н	1	1
Н.контр.		Демська Н.П.				Кафедра КІТАР ХНУРЕ		
Затв.		Невлюдов І.Ш.						