Додаток В

Лістинг програми

Altimeter_Vertical_Speed.ino

```
/*
 * Програмне забезпечення №2
 * Це програмне забезпечення розроблено для дипломного проекту
 * "Прилад для моніторингу польотної інформації"
 * Код розроблено для мікроконтролера Atmega238P
 * Дисплей для відображення графіки розширенням 320x240 на
 * мікросхемі ST7789
 * Датчик тиску HSCMAND015PA2A3
 */


#include <Adafruit_GFX.h>    // Бібліотека для відображення графіки
#include <Adafruit_ST7789.h> // Бібліотека для роботи із дисплеєм
#include <SPI.h>             // Бібліотека для роботи інтерфейсу SPI
#include <Wire.h>
#include <SSC.h>             // Бібліотека для роботи з датчиком тиску
#include <TimerOne.h>        // Бібліотека для створення переривань


// Налаштування ініціалізація


#define TFT_CS    10
#define TFT_DC     9
#define TFT_RST    8


Adafruit_ST7789 tft = Adafruit_ST7789(TFT_CS, TFT_DC, TFT_RST);
SSC ssc(0x28, 8);


const int encoderPinA = 2;
```

```
const int encoderPinB = 3;
const int buttonPin = 4;

volatile int counter = 1013;
volatile bool buttonPressed = false;

int ALT = 0;
int prev_ALT = 0;

int V_SPEED = 0;
int prev_V_SPEED = 0;

int PSI = 0;
int prev_PSI = 0;

unsigned long prevTime = 0;  // Змінна для зберігання попереднього часу
const unsigned long interval = 1000;  // Інтервал часу (1 секунда)

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

  ssc.setMinRaw(1598);
  ssc.setMaxRaw(14701);
  ssc.setMinPressure(0.0);
  ssc.setMaxPressure(1500);
  ssc.update();

  // Початкова сторінка при завантажені
  tft.init(240, 320, SPI_MODE2);
  tft.setRotation(3);

  tft.fillScreen(ST77XX_BLACK);
```

```
  tft.setTextSize(2);
  tft.setCursor(45, 50);
  tft.setTextColor(ST77XX_WHITE);
  tft.setTextWrap(true);
  tft.print("BAROMETRIC ALTIMETER");

  tft.setTextSize(2);
  tft.setCursor(45, 80);
  tft.setTextColor(ST77XX_WHITE);
  tft.setTextWrap(true);
  tft.print("from -1000 to 5000 m");

  tft.setCursor(80, 130);
  tft.setTextColor(ST77XX_WHITE);
  tft.setTextWrap(true);
  tft.print("VERTICAL SPEED");

  tft.setTextSize(2);
  tft.setCursor(60, 170);
  tft.setTextColor(ST77XX_WHITE);
  tft.setTextWrap(true);
  tft.print("from -30 to 30 m/s");

  delay(5000);
  tft.fillScreen(ST77XX_BLACK);

  pinMode(encoderPinA, INPUT);
  pinMode(encoderPinB, INPUT);
  pinMode(buttonPin, INPUT_PULLUP);

  attachInterrupt(digitalPinToInterrupt(encoderPinA),
handleEncoder, CHANGE);
  attachInterrupt(digitalPinToInterrupt(buttonPin), handleButton,
FALLING);
```

```
}


float pressure = 0;

float T0 = 288.15; // температура на рівні моря (у Кельвінах). Зазвичай приймається приблизно 288.15 К.

float L = 0.0065; // температурний градієнт атмосфери. Це зміна температури з висотою. Зазвичай приймається приблизно 0.0065 К/м.

float R = 8.314;  // газова стала. Зазвичай приймається приблизно 8.314 J/(mol·K).

float g = 9.80665; // прискорення вільного падіння. Зазвичай приймається приблизно 9.80665 м/с².


void loop()

{

  unsigned long currentTime = millis();


  // Оновлення даних тиску над рівнем моря задане Енкодером

  PSI = counter;


  // Оновлення даних із датчика тиску

  ssc.update();

  pressure = ssc.pressure();


  // Розрахунок барометричної висоти

  ALT = (T0 / L) * (1 - pow((pressure / PSI),(R*L/g)));


  // Оновлення даних на диплеї

  if (currentTime - prevTime >= interval)

  {

    ssc.update();

    pressure = ssc.pressure();

    V_SPEED = (T0 / L) * (1 - pow((pressure / PSI),(R*L/g))) - ALT;

    prevTime = currentTime;
```

```
  }

  // Виведення даних на дисплей
  ALT_VSPEED_indicator();

}

// Обробник переривань Енкодера
void handleEncoder()
{
  static byte oldState = 0;
  byte newState = (digitalRead(encoderPinB) << 1) |
digitalRead(encoderPinA);
  byte t = oldState << 2 | newState;

  switch (t)
  {
    case 0b0001:
    case 0b0111:
    case 0b1110:
    case 0b1000:
      counter++;
      break;
    case 0b0010:
    case 0b0100:
    case 0b1101:
    case 0b1011:
      counter--;
      break;
  }

  oldState = newState;
}
```

```
// Обробка переривань кнопки Енкодера
void handleButton()
{
  buttonPressed = true;
}

// Відображення поділок індикатора вертикальної швидкості
void ALT_VSPEED_indicator()
{
   tft.setTextSize(2);
   tft.setCursor(160, 15);
   tft.setTextColor(ST77XX_WHITE);
   tft.setTextWrap(true);
   tft.print("ALTIMETR");

   tft.setTextSize(2);
   tft.setCursor(10, 15);
   tft.setTextColor(ST77XX_WHITE);
   tft.setTextWrap(true);
   tft.print("V_SPEED");

   tft.drawRect(110, 50, 210, 140, ST77XX_WHITE);

   tft.setTextColor(ST77XX_WHITE);
   tft.setTextSize(6);
   tft.setCursor(275, 105);
   tft.print("m");

   tft.setTextColor(ST77XX_CYAN);
   tft.setTextSize(2);
   tft.setCursor(265, 215);
   tft.print("PSI");
```

```
   tft.setTextColor(ST77XX_MAGENTA);
   tft.setTextSize(2);
   tft.setCursor(100, 215);
   tft.print("m/s");

    for(int i = -6; i <= 6; i++)
    {
      if(i % 2)
      {
        tft.drawLine(5, 120 + i * 10, 15, 120 + i * 10,
ST77XX_MAGENTA);
      }
      else
      {
        tft.drawLine(5, 120 + i * 10, 20, 120 + i * 10,
ST77XX_MAGENTA);
        tft.setTextColor(ST77XX_MAGENTA);
        tft.setTextSize(1);
        tft.setCursor(28, 118 + i * 10);
        tft.print(abs(i * 5));
      }
    }

   if (ALT != prev_ALT)
   {
      tft.setTextColor(ST77XX_BLACK);
      tft.setTextSize(6);
      tft.setCursor(120, 105);
      tft.print(prev_ALT);

      prev_ALT = ALT;

      tft.setTextColor(ST77XX_WHITE);
      tft.setTextSize(6);
```

```
      tft.setCursor(120, 105);
      tft.print(ALT);
   }

   if (V_SPEED != prev_V_SPEED)
   {
      tft.setTextColor(ST77XX_BLACK);
      tft.setTextSize(4);
      tft.setCursor(20, 200);
      tft.print(prev_V_SPEED);

      tft.fillTriangle(25,120 - prev_V_SPEED * 2,45,115 -
prev_V_SPEED * 2,45,125 - prev_V_SPEED * 2, ST77XX_BLACK);

      prev_V_SPEED = V_SPEED;

      tft.fillTriangle(25,120 - V_SPEED * 2,45,115 - V_SPEED *
2,45,125 - V_SPEED * 2, ST77XX_YELLOW);

      tft.setTextColor(ST77XX_MAGENTA);
      tft.setTextSize(4);
      tft.setCursor(20, 200);
      tft.print(V_SPEED);
   }

   if (PSI != prev_PSI)
   {
     tft.setTextColor(ST77XX_BLACK);
     tft.setTextSize(4);
     tft.setCursor(155, 200);
     tft.print(prev_PSI);

     prev_PSI = PSI;
```

```
      tft.setTextColor(ST77XX_CYAN);
      tft.setTextSize(4);
      tft.setCursor(155, 200);
      tft.print(PSI);
    }

}
```