Додаток Б

Лістинг програми

Air_horizon_module.ino

```
/*
 * Програмне забезпечення №1
 * Це програмне забезпечення розроблено для дипломного проєкту
 * "Прилад для моніторингу польотної інформації"
 * Код розроблено для мікроконтролера Atmega238P
 * Дисплей для відображення графіки розширенням 320x240 на мікросхемі ST7789
 * Гіроскоп на мікросхемі MPU6050
 */

#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 "I2Cdev.h"           // Бібліотека для роботи інтерфейсу I2C

// Налаштування та ініціалізація

#include "MPU6050_6Axis_MotionApps20.h"

#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
    #include "Wire.h"
#endif

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

```
MPU6050 mpu;
Adafruit_ST7789 tft = Adafruit_ST7789(TFT_CS, TFT_DC, TFT_RST);


int ROLL = 0;
int PITCH = 0;
int prev_ROLL = 0;
int prev_PITCH = 0;
int prev2_PITCH = 0;

int cord_date[8] = {0,0,0,0,0,0,0,0};
int ind_date[6] = {0,0,0,0,0,0};

#define INTERRUPT_PIN 2
#define LED_PIN 13
bool blinkState = false;

bool dmpReady = false;
uint8_t mpuIntStatus;
uint8_t devStatus;
uint16_t packetSize;
uint16_t fifoCount;
uint8_t fifoBuffer[64];

Quaternion q;
VectorInt16 aa;
VectorInt16 aaReal;
VectorInt16 aaWorld;
VectorFloat gravity;
float euler[3];
float ypr[3];

uint8_t teapotPacket[14] = { '$',
```

```
                                0x02,
                                0,0,
                                0,0,
                                0,0,
                                0,0,
                                0x00,
                                0x00,
                                '\r',
                                '\n' };

volatile bool mpuInterrupt = false;

void dmpDataReady() {
    mpuInterrupt = true;
}

void setup()
{
    #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
        Wire.begin();
        Wire.setClock(400000);
    #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
        Fastwire::setup(400, true);
    #endif

    Serial.begin(115200);
    while (!Serial);
    mpu.initialize();
    pinMode(INTERRUPT_PIN, INPUT);
    devStatus = mpu.dmpInitialize();

    mpu.setXGyroOffset(220);
    mpu.setYGyroOffset(76);
```

```
  mpu.setZGyroOffset(-85);
  mpu.setZAccelOffset(1788);

  if (devStatus == 0) {
      mpu.CalibrateAccel(6);
      mpu.CalibrateGyro(6);
      mpu.PrintActiveOffsets();
      mpu.setDMPEnabled(true);
      Serial.print(digitalPinToInterrupt(INTERRUPT_PIN));
      attachInterrupt(digitalPinToInterrupt(INTERRUPT_PIN),
      dmpDataReady, RISING);
      mpuIntStatus = mpu.getIntStatus();
      dmpReady = true;
      packetSize = mpu.dmpGetFIFOPacketSize();
  } else {
      // ERROR!
      // 1 = initial memory load failed
      // 2 = DMP configuration updates failed
      // (if it's going to break, usually the code will be 1)
      Serial.print(F("Initialization failed (code "));
      Serial.print(devStatus);
      Serial.println(F(")"));
  }
  pinMode(LED_PIN, OUTPUT);

// Початкова сторінка при завантаженні
tft.init(240, 320, SPI_MODE2);
tft.setRotation(3);
tft.fillScreen(ST77XX_BLACK);
tft.setTextSize(2);
tft.setCursor(30, 50);
tft.setTextColor(ST77XX_WHITE);
tft.setTextWrap(true);
```

```
  tft.print("Aviahorizont module V1");
  icon_plane();
  delay(5000);
  tft.fillScreen(ST77XX_BLACK);
  calculate_ROLL();
}

void loop()
{
  // Відоюраження цифрового поля Тангаж
  Text_ROLL();
  // Відображення цифрового поля Крен
  Text_PITCH();

  // Відображення Авіагоризонту
  calculate_ROLL();
  // Відображення кординатора Тангаж
  calculate_ROLL_indicator();
  // Відображення поділок Крен
  calculate_PITCH_indicator();

  // Відображення статичної іконки літака
  icon_plane();

  // Отримання даних із гіроскопа
  if (!dmpReady) return;
  if (mpu.dmpGetCurrentFIFOPacket(fifoBuffer))

  // Оновлення даних із гіроскопа
  mpu.dmpGetQuaternion(&q, fifoBuffer);
  mpu.dmpGetGravity(&gravity, &q);
  mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);
```

```
  // Перерахунок даних у кут нахилу
  ROLL = -(ypr[0] * 180/M_PI);
  PITCH = ypr[2] * 180/M_PI;

}

// Відображення поділок Крен
void calculate_PITCH_indicator()
{
   if (PITCH != prev2_PITCH)
  {
    for(int i = -3; i <= 3; i++)
    {
      tft.drawLine(120, 120 + prev2_PITCH * 2 + i * 20, 200, 120 +
prev2_PITCH * 2 + i * 20, ST77XX_BLACK);

      tft.setTextColor(ST77XX_BLACK);
      tft.setTextSize(1);
      tft.setCursor(100, 120 + prev2_PITCH * 2 + i * 20);
      tft.print(abs(i * 10));

      tft.setTextColor(ST77XX_BLACK);
      tft.setTextSize(1);
      tft.setCursor(215, 120 + prev2_PITCH * 2 + i * 20);
      tft.print(abs(i * 10));
    }

    prev2_PITCH = PITCH;

    for(int i = -3; i <= 3; i++)
    {
      tft.drawLine(120, 120 + PITCH * 2 + i * 20, 200, 120 + PITCH
* 2 + i * 20, ST77XX_WHITE);
```

```
      tft.setTextColor(ST77XX_WHITE);
      tft.setTextSize(1);
      tft.setCursor(100, 120 + PITCH * 2 + i * 20);
      tft.print(abs(i * 10));

      tft.setTextColor(ST77XX_WHITE);
      tft.setTextSize(1);
      tft.setCursor(215, 120 + PITCH * 2 + i * 20);
      tft.print(abs(i * 10));
    }
  }
}

// Відоюраження цифрового поля Тангаж
void Text_ROLL()
{
  if (ROLL != prev_ROLL)
  {
    tft.setTextColor(ST77XX_BLACK);
    tft.setTextSize(1);
    tft.setCursor(10, 200);
    tft.print("ROLL: ");
    tft.setTextSize(2);
    tft.setCursor(10, 210);
    tft.print(abs(prev_ROLL));

    prev_ROLL = ROLL;

    tft.setTextColor(ST77XX_WHITE);
    tft.setTextSize(1);
    tft.setCursor(10, 200);
    tft.print("ROLL: ");
    tft.setTextSize(2);
```

```
    tft.setCursor(10, 210);
    tft.print(abs(ROLL));
  }
}

// Відображення цифрового поля Крен
void Text_PITCH()
{
   if (PITCH != prev_PITCH)
  {
    tft.setTextColor(ST77XX_BLACK);
    tft.setTextSize(1);
    tft.setCursor(270, 200);
    tft.print("PITCH: ");
    tft.setTextSize(2);
    tft.setCursor(270, 210);
    tft.print(prev_PITCH);

    prev_PITCH = PITCH;

    tft.setTextColor(ST77XX_WHITE);
    tft.setTextSize(1);
    tft.setCursor(270, 200);
    tft.print("PITCH: ");
    tft.setTextSize(2);
    tft.setCursor(270, 210);
    tft.print(PITCH);
  }
}

// Відображення статичної іконки літака
void icon_plane()
{
```

```
  tft.fillTriangle(160, 120, 210, 150, 160, 140, ST77XX_YELLOW);
  tft.fillTriangle(160, 120, 110, 150, 160, 140, ST77XX_YELLOW);
}

// Відображення Авіагоризонту
void calculate_ROLL()
{

tft.drawLine(cord_date[0],cord_date[1],cord_date[2],cord_date[3],
ST77XX_BLACK);

tft.drawLine(cord_date[4],cord_date[5],cord_date[6],cord_date[7],
ST77XX_BLACK);


  float theta_roll_a = 2.0f * 3.1415926f * float(ROLL) /
float(360);
  int x_roll_a = (600 * cosf(theta_roll_a)) + 160;
  int y_roll_a = (600 * sinf(theta_roll_a)) + 120 + PITCH * 2;


  float theta_roll_b = 2.0f * 3.1415926f * float(ROLL + 180) /
float(360);
  int x_roll_b = (600 * cosf(theta_roll_b)) + 160;
  int y_roll_b = (600 * sinf(theta_roll_b)) + 120 + PITCH * 2;


  cord_date[0] = x_roll_a;
  cord_date[1] = y_roll_a;
  cord_date[2] = x_roll_b;
  cord_date[3] = y_roll_b;



tft.drawLine(cord_date[0],cord_date[1],cord_date[2],cord_date[3],
ST77XX_ORANGE);
}

// Відображення кординатора Тангаж
void calculate_ROLL_indicator()
```

```
{
  tft.setTextSize(1);
  tft.setCursor(70, 60);
  tft.setTextColor(ST77XX_WHITE);
  tft.setTextWrap(true);
  tft.print("50");

  tft.setCursor(240, 60);
  tft.setTextColor(ST77XX_WHITE);
  tft.setTextWrap(true);
  tft.print("50");

  for(int i = 220; i < 320; i++)
  {
    float theta_a = 2.0f * 3.1415926f * float(i) / float(360);
    int x_a = (110 * cosf(theta_a)) + 160;
    int y_a = (110 * sinf(theta_a)) + 120;

    float theta_b = 2.0f * 3.1415926f * float(i) / float(360);
    int x_b = (110 * cosf(theta_b)) + 160;
    int y_b = (110 * sinf(theta_b)) + 120;

    tft.drawLine(x_a, y_a, x_b, y_b, ST77XX_WHITE);
  }

  for(int i = 220; i <= 320; i += 10)
  {
    float theta_a = 2.0f * 3.1415926f * float(i) / float(360);
    int x_a = (110 * cosf(theta_a)) + 160;
    int y_a = (110 * sinf(theta_a)) + 120;

    float theta_b = 2.0f * 3.1415926f * float(i) / float(360);
    int x_b = (100 * cosf(theta_b)) + 160;
```

```
    int y_b = (100 * sinf(theta_b)) + 120;


    tft.drawLine(x_a, y_a, x_b, y_b, ST77XX_WHITE);
  }


  if(ROLL >= -50 && ROLL <= 50)
  {

tft.fillTriangle(ind_date[0],ind_date[1],ind_date[2],ind_date[3],i
nd_date[4],ind_date[5], ST77XX_BLACK);


    float theta_a = 2.0f * 3.1415926f * float(265 + ROLL) /
float(360);
    int x_a = (90 * cosf(theta_a)) + 160;
    int y_a = (90 * sinf(theta_a)) + 120;


    float theta_b = 2.0f * 3.1415926f * float(270 + ROLL) /
float(360);
    int x_b = (105 * cosf(theta_b)) + 160;
    int y_b = (105 * sinf(theta_b)) + 120;


    float theta_c = 2.0f * 3.1415926f * float(275 + ROLL) /
float(360);
    int x_c = (90 * cosf(theta_c)) + 160;
    int y_c = (90 * sinf(theta_c)) + 120;


    ind_date[0] = x_a;
    ind_date[1] = y_a;
    ind_date[2] = x_b;
    ind_date[3] = y_b;
    ind_date[4] = x_c;
    ind_date[5] = y_c;


tft.fillTriangle(ind_date[0],ind_date[1],ind_date[2],ind_date[3],i
nd_date[4],ind_date[5], ST77XX_WHITE);
```

```
    tft.fillRect(65, 210, 190, 230, ST77XX_BLACK);
  }
  else
  {

tft.fillTriangle(ind_date[0],ind_date[1],ind_date[2],ind_date[3],i
nd_date[4],ind_date[5], ST77XX_RED);
    tft.fillRect(65, 210, 190, 230, ST77XX_RED);
    tft.setTextSize(2);
    tft.setCursor(75, 220);
    tft.setTextColor(ST77XX_WHITE);
    tft.setTextWrap(true);
    tft.print("LEVEL THE ROLL!");
  }
}
```