ДОДАТОК А
Програмна реалізація апаратної частини

```
#include <Wire.h>
#include <SoftwareSerial.h>
#include <GyverButton.h>
#include "I2Cdev.h"
SoftwareSerial BTserial(2, 3);

const int MPU = 0x68; // MPU6050 I2C address
float AccX, AccY, AccZ;
float GyroX, GyroY, GyroZ;
float accAngleX, accAngleY, gyroAngleX, gyroAngleY, gyroAngleZ;
float roll, pitch, yaw;
float AccErrorX = 0.6;
float AccErrorY = -3.12;
float GyroErrorX = -2.23;
float GyroErrorY = 2.39;
float GyroErrorZ = 0.66;
float elapsedTime, currentTime, previousTime;
int c = 0;
GButton B_Reset(5);
GButton B_FingerLeft(6);
GButton B_FingerRight(7);
unsigned int mpuTimer = 0;
unsigned int resetTimer = 0;
unsigned int leftClickTimer = 0;
unsigned int rightClickTimer = 0;
bool isLeftHolded = 0;
bool isRightHolded = 0;

void setup() {
```

```
  Serial.begin(9600);
  BTserial.begin(9600);
  Wire.begin();                  // Initialize comunication
  Wire.beginTransmission(MPU);       // Start communication with MPU6050 // MPU=0x68
  Wire.write(0x6B);                  // Talk to the register 6B
  Wire.write(0x00);                  // Make reset - place a 0 into the 6B register
  Wire.endTransmission(true);        //end the transmission
    // get the IMU error values for your module
  calculate_IMU_error();
  delay(20);
  mpuTimer = resetTimer = leftClickTimer = rightClickTimer = millis();
  B_FingerLeft.setTimeout(100);
  B_FingerRight.setTimeout(100);
}
void loop() {
  B_Reset.tick();
  B_FingerLeft.tick();
  B_FingerRight.tick();

  RightClickHandler();
  LeftClickHandler();
  ResetButtonHandler();

  if(millis() - mpuTimer > 30){
    mpuTimer = millis();
    SendMPU6050Data();
  }
}

void RightClickHandler(){
  if(B_FingerRight.isHold()){
    isRightHolded = 1;
```

```
  }
  else{
    isRightHolded = 0;
  }
}
void LeftClickHandler(){
  if(B_FingerLeft.isHold()){
    isLeftHolded = 1;
  }
  else{
    isLeftHolded = 0;
  }
}
void ResetButtonHandler(){
  if(B_Reset.isPress()){
    Wire.begin();
    Wire.beginTransmission(MPU);
    Wire.write(0x6B);
    Wire.write(0x00);
    Wire.endTransmission(true);
    accAngleX = 0;
    accAngleY = 0;
    gyroAngleX = 0;
    gyroAngleY = 0;
    gyroAngleZ = 0;
    yaw =0;
    Serial.println("Reset");
    return;
  }
}
void SendMPU6050Data(){
  // === Read acceleromter data === //
```

```
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); // Start with register 0x3B (ACCEL_XOUT_H)
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); // Read 6 registers total, each axis value is stored in 2 registers
  //For a range of +-2g, we need to divide the raw values by 16384, according to the datasheet
  AccX = (Wire.read() << 8 | Wire.read()) / 16384.0; // X-axis value
  AccY = (Wire.read() << 8 | Wire.read()) / 16384.0; // Y-axis value
  AccZ = (Wire.read() << 8 | Wire.read()) / 16384.0; // Z-axis value
  // Calculating Roll and Pitch from the accelerometer data
  accAngleX = (atan(AccY / sqrt(pow(AccX, 2) + pow(AccZ, 2))) * 180 / PI) - AccErrorX; // AccErrorX ~(0.58) See the
calculate_IMU_error()custom function for more details
  accAngleY = (atan(-1 * AccX / sqrt(pow(AccY, 2) + pow(AccZ, 2))) * 180 / PI) - AccErrorY; // AccErrorY ~(-1.58)
  // === Read gyroscope data === //
  previousTime = currentTime;        // Previous time is stored before the actual time read
  currentTime = millis();            // Current time actual time read
  elapsedTime = (currentTime - previousTime) / 1000; // Divide by 1000 to get seconds
  Wire.beginTransmission(MPU);
  Wire.write(0x43); // Gyro data first register address 0x43
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); // Read 4 registers total, each axis value is stored in 2 registers
  GyroX = (Wire.read() << 8 | Wire.read()) / 131.0; // For a 250deg/s range we have to divide first the raw value by
131.0, according to the datasheet
  GyroY = (Wire.read() << 8 | Wire.read()) / 131.0;
  GyroZ = (Wire.read() << 8 | Wire.read()) / 131.0;
    // Correct the outputs with the calculated error values
  GyroX = GyroX - GyroErrorX;
  GyroY = GyroY - GyroErrorY;
  GyroZ = GyroZ - GyroErrorZ;
  // Currently the raw values are in degrees per seconds, deg/s
  gyroAngleX = gyroAngleX + GyroX * elapsedTime; // deg/s * s = deg
  gyroAngleY = gyroAngleY + GyroY * elapsedTime;
  yaw =  yaw + GyroZ * elapsedTime;
```

```
  // Complementary filter - combine acceleromter and gyro angle values
  roll = 0.96 * gyroAngleX + 0.04 * accAngleX;
  pitch = 0.96 * gyroAngleY + 0.04 * accAngleY;
  Serial.println((String)roll + "/" + pitch + "/" + yaw );
  BTserial.println((String)roll + " " + pitch + " " + yaw + " " + isLeftHolded + " " + isRightHolded);
}
void calculate_IMU_error() {
  while (c < 200) {
    Wire.beginTransmission(MPU);
    Wire.write(0x3B);
    Wire.endTransmission(false);
    Wire.requestFrom(MPU, 6, true);
    AccX = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
    AccY = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
    AccZ = (Wire.read() << 8 | Wire.read()) / 16384.0 ;
    // Sum all readings
    AccErrorX = AccErrorX + ((atan((AccY) / sqrt(pow((AccX), 2) + pow((AccZ), 2))) * 180 / PI));
    AccErrorY = AccErrorY + ((atan(-1 * (AccX) / sqrt(pow((AccY), 2) + pow((AccZ), 2))) * 180 / PI));
    c++;
  }
  AccErrorX = AccErrorX / 200;
  AccErrorY = AccErrorY / 200;
  c = 0;
  // Read gyro values 200 times
  while (c < 200) {
    Wire.beginTransmission(MPU);
    Wire.write(0x43);
    Wire.endTransmission(false);
    Wire.requestFrom(MPU, 6, true);
    GyroX = Wire.read() << 8 | Wire.read();
    GyroY = Wire.read() << 8 | Wire.read();
    GyroZ = Wire.read() << 8 | Wire.read();
```

```
    // Sum all readings
    GyroErrorX = GyroErrorX + (GyroX / 131.0);
    GyroErrorY = GyroErrorY + (GyroY / 131.0);
    GyroErrorZ = GyroErrorZ + (GyroZ / 131.0);
    c++;
  }
  //Divide the sum by 200 to get the error value
  GyroErrorX = GyroErrorX / 200;
  GyroErrorY = GyroErrorY / 200;
  GyroErrorZ = GyroErrorZ / 200;
  // Print the error values on the Serial Monitor
  Serial.print("AccErrorX: ");
  Serial.println(AccErrorX);
  Serial.print("AccErrorY: ");
  Serial.println(AccErrorY);
  Serial.print("GyroErrorX: ");
  Serial.println(GyroErrorX);
  Serial.print("GyroErrorY: ");
  Serial.println(GyroErrorY);
  Serial.print("GyroErrorZ: ");
  Serial.println(GyroErrorZ);
}
```

ДОДАТОК Б
Програмна реалізація комп’ютерного додатку

```
public partial class MainForm : Form
 {
  private SerialPort _serialPort;
  private Point _cursorPosition;
  private Thread _readThread;
  public MainForm()
  {
   InitializeComponent();
  }
  private void Form_Load(object sender, EventArgs e)
  {
   RefreshSerialPorts();
   _readThread = new Thread(Test);
   _readThread.Start();
  }
  private void t_Refresh_Tick(object sender, EventArgs e)
  {
   pb_Canvas.Refresh();
  }
  private void pb_Canvas_Paint(object sender, PaintEventArgs e)
  {
   e.Graphics.DrawEllipse(Pens.Brown, _cursorPosition.X, _cursorPosition.Y, 10, 10);
   e.Graphics.FillEllipse(Brushes.Brown, _cursorPosition.X, _cursorPosition.Y, 10, 10);
  }

  private void MainFormClosed(object sender, FormClosedEventArgs e)
  {
   _serialPort?.Dispose();
  }
```

```
private void pb_Canvas_MouseMove(object sender, MouseEventArgs e)
{
  _cursorPosition = e.Location;
}
private void tb_Distance_Scroll(object sender, EventArgs e)
{
  nud_Distance.Value = tb_Distance.Value;
}
private void nud_Distance_ValueChanged(object sender, EventArgs e)
{
  tb_Distance.Value = (int) nud_Distance.Value;
}
private void b_Connect_Click(object sender, EventArgs e)
{
  const string connectText = "Connect";
  const string disconnectText = "Disconnect";

  if (b_Connect.Text.Equals(connectText))
  {
    SerialPortInit();
    b_Connect.Text = disconnectText;
  }
  else
  {
    _serialPort?.Dispose();
    b_Connect.Text = connectText;
  }
}
private void cb_SerialPort_DropDown(object sender, EventArgs e)
{
  RefreshSerialPorts();
}
```

```
private void SerialPortInit()
{
  var portName = (string) cb_SerialPort.SelectedItem;
  //ToDo: verify long work period (stop work after period of time near 5 min)
  var port = new SerialPort
  {
    PortName = portName,
    BaudRate = 9600,
    Parity = Parity.None,
    DataBits = 8,
    StopBits = StopBits.One,
    Handshake = Handshake.None,
  };
  try
  {
    port.Open();
  }
  catch (Exception ex)
  {
    MessageBox.Show(ex.ToString());
    return;
  }
  _readThread = new Thread(ReadPortOnDataReceived);
  _readThread.Start(port);
  _serialPort = port;
}
private void ReadPortOnDataReceived(object portObject)
{
  var serialPort = portObject as SerialPort;
  if (serialPort is null)
  {
    MessageBox.Show("PortObject must be a SerialPort class object");
```

```
    throw new ArgumentException("PortObject must be a SerialPort class object");
  }
  var mouseService = new MouseService();
  var sw = new Stopwatch();
  while (serialPort.IsOpen)
  {
    sw.Restart();
    var bluetoothDto = _serialPort.ReadControllerData();
    if (bluetoothDto is null)
    {
      continue;
    }
    var coordinate = bluetoothDto.Angles.ToCoordinate((int) nud_Distance.Value);
    mouseService.MoveAndClick(new MouseCommandModel(
      coordinate,
      bluetoothDto.LeftButtonState == 1,
      bluetoothDto.RightButtonState == 1));
    sw.Stop();
    Console.WriteLine($"{sw.ElapsedMilliseconds}");
  }
}
private void RefreshSerialPorts()
{
  var ports = SerialPort.GetPortNames().OrderBy(p => p).ToArray();
  if (!ports.Any())
  {
    MessageBox.Show("There is no serial ports to show. Pair Arduino controller by Bluetooth");
    throw new ArgumentException("There is no serial ports to show");
  }
  var selectedItem = cb_SerialPort.SelectedItem as string;
  cb_SerialPort.Items.Clear();
  cb_SerialPort.Items.AddRange(ports);
```

```
    cb_SerialPort.SelectedItem = selectedItem != null && ports.Contains(selectedItem)
      ? selectedItem
      : cb_SerialPort.Items[0];
  }
  private void Test()
  {
    var mouseService = new MouseService();
    for (var i = 0; i < 10; i++)
    {
      var x = 1000 + 50 * i;
      var mouseCommand = new MouseCommandModel(new Point(x, 0), false, false);
      mouseService.MoveAndClick(mouseCommand);
      Console.WriteLine(x);
      Thread.Sleep(1000);
    }
  }
 }
public class MouseService
 {
  private bool IsLeftMouseButtonPressed { get; set; }
  private bool IsRightMouseButtonPressed { get; set; }
  public void MoveAndClick(MouseCommandModel command)
  {
    var screenSize = Screen.PrimaryScreen.Bounds.Size;
    var center = new Point(screenSize.Width / 2, screenSize.Height / 2);
    var x = center.X + command.MousePosition.X;
    var y = center.Y + command.MousePosition.Y;
    Cursor.Position = new Point(x, y);
    //press if released
    if (IsLeftMouseButtonPressed != command.PressLeftButton)
    {
      var mouseEvent = command.PressLeftButton ? MouseFlags.LeftDown : MouseFlags.LeftUp;
```

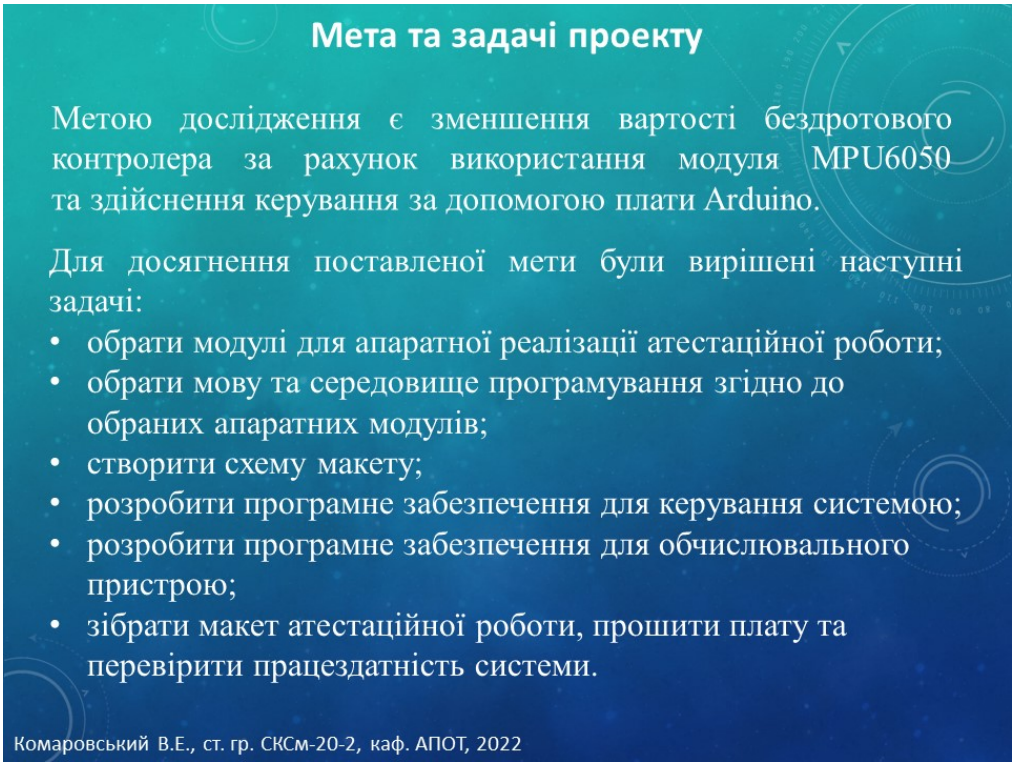
```
    MakeMouseEvent(mouseEvent);
    IsLeftMouseButtonPressed = !IsLeftMouseButtonPressed;
    var action = mouseEvent == MouseFlags.LeftDown ? "Pressed" : "Released";
    Console.Write($"Left {action}");
   }

   if (IsRightMouseButtonPressed != command.PressRightButton)
   {
    var mouseEvent = command.PressRightButton ? MouseFlags.RightDown : MouseFlags.RightUp;
    MakeMouseEvent(mouseEvent);
    IsRightMouseButtonPressed = !IsRightMouseButtonPressed;
    var action = mouseEvent == MouseFlags.LeftDown ? "Pressed" : "Released";
    Console.Write($"Right {action}");
   }
  }
  [DllImport("User32.dll")]
  private static extern void mouse_event(MouseFlags dwFlags, int dx, int dy, int dwData, UIntPtr dwExtraInfo);
  [Flags]
  enum MouseFlags
  {
   Move = 0x0001,
   LeftDown = 0x0002,
   LeftUp = 0x0004,
   RightDown = 0x0008,
   RightUp = 0x0010,
   Absolute = 0x8000,
   MiddleDown = 0x0020,
   MiddleUp = 0x0040,
   XDown = 0x0080,
   XUp = 0x0100,
   Wheel = 0x0800,
   HWheel = 0x01000, // Horizontal wheel
```

```
    }
    private static void MakeMouseEvent(MouseFlags mouseEvent)
    {
      mouse_event(mouseEvent, Cursor.Position.X, Cursor.Position.Y, 0, UIntPtr.Zero);
    }
  }
public class BluetoothDto
  {
    public EulerAngles Angles { get; }
    public int LeftButtonState { get; }
    public int RightButtonState { get; }

    public BluetoothDto(EulerAngles angles, int leftButtonState, int rightButtonState)
    {
      Angles = angles;
      LeftButtonState = leftButtonState;
      RightButtonState = rightButtonState;
    }
  }
public struct EulerAngles
  {
    public float X; // Roll
    public float Y; // Pitch
    public float Z; // Yaw

    public EulerAngles(float x, float y, float z)
    {
      X = x;
      Y = y;
      Z = z;
    }
  }
```

```
public class MouseCommandModel
  {
    public Point MousePosition { get; }
    public bool PressLeftButton { get; }
    public bool PressRightButton { get; }
    public MouseCommandModel(Point mousePosition, bool pressLeftButton, bool pressRightButton)
    {
      MousePosition = mousePosition;
      PressLeftButton = pressLeftButton;
      PressRightButton = pressRightButton;
    }
  }
public static class EulerAngelsExtensions
  {
    /// <summary>
    /// Get x,y coordinates of ray point projection on the screen
    /// </summary>
    /// <param name="angles"> Euler Angles</param>
    /// <param name="h"> DistanceToScreen</param>
    /// <returns></returns>
    public static Point ToCoordinate(this EulerAngles angles, float h)
    {
      var xProjection = GetProjection(angles.X, h);
      var yProjection = GetProjection(angles.Y, h);

      var x = xProjection * GetSign(angles.X);
      var y = yProjection * GetSign(angles.Y);
      return new Point((int) x, (int) y);
    }
    private static double GetProjection(float angel, float h)
    {
      var absAngle = Math.Abs(angel);
```

```
    var sinAlfa = Math.Sin(GetRadian(absAngle));
    var sinBetta = Math.Sin(GetRadian(90 - absAngle)); // 180 - 90 - angle
    return sinAlfa * h / sinBetta;
   }
   private static int GetSign(float angle) => angle > 0 ? 1 : -1;
   private static double GetRadian(double degrees) => Math.PI * degrees / 180.0;
  }
public static class SerialPortExtensions
  {
   public static BluetoothDto ReadControllerData(this SerialPort port)
   {
    var data = port.ReadLine().Trim('\r');
    if (string.IsNullOrWhiteSpace(data))
    {
     return null;
    }
    //validation: is all params given
    var readParams = data.Split(' ', StringSplitOptions.RemoveEmptyEntries);
    if (readParams.Length != 5 ||
        !float.TryParse(readParams[0].Replace('.', ','), out var x) ||
        !float.TryParse(readParams[1].Replace('.', ','), out var y) ||
        !float.TryParse(readParams[2].Replace('.', ','), out var z) ||
        !int.TryParse(readParams[3], out var isLeftHolded) ||
        !int.TryParse(readParams[4], out var isRightHolded))
    {
     return null;
    }
    // invert param to prevent axis inversion
    return new BluetoothDto(new EulerAngles(-x, y, -z), isLeftHolded, isRightHolded);
   }
  }
```

ДОДАТОК В
Графічний матеріал



Актуальність

З розвитком нових технологій виникає все більше галузей, які мають не стандартний тип інтерфейсу порівняно з персональним комп'ютером, тому виникає потреба у нових видах контролерів для роботи з такими обчислювальними пристроями.

Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Переваги

До головних переваг можна віднести:

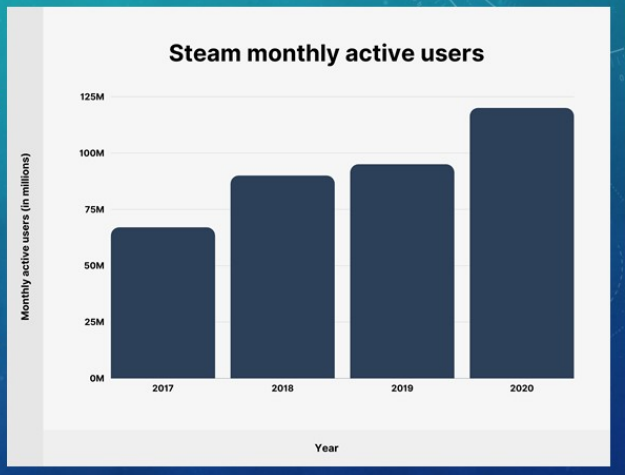
- ціна;
- портативність;
- відкритий програмний код;
- зручність в обслуговуванні;
- зручність в оновленні кодової прошивки.

Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Затребуваність

За деякими даними частина VR у секторі розваг стрімко зростає останнім часом. За перший квартал 2021 рок цей ріст становить 52.4%.

За даними загально відомої ігрової платформи Steam понад 2% користувачів використовують VR технології. При заявлених 120 млн активних користувачів на місяць 2% становлять приблизно 2,4 млн.



Вибір апаратних модулів

Розроблений макет атестаційної роботи повинен відповідати наступним вимогам:

- надійність;
- з’єднуватись з обчислювальним пристроєм за допомогою технології Bluetooth;
- можливість швидко та легко змінювати прошивку приладу;
- невеликі витрати електроенергії;
- компактність – в результаті прилад не повинен мати великих габаритів;
- мобільність – кінцева розробка з легкістю переміщується.

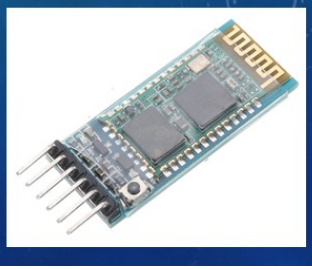
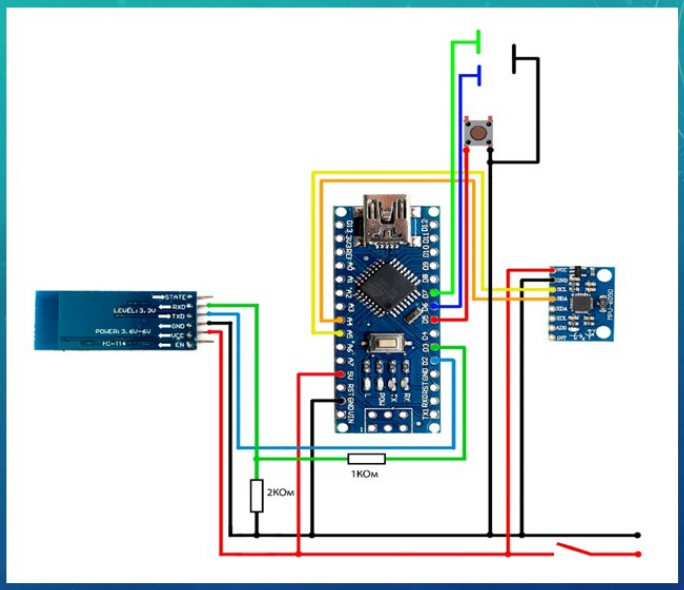


Схема макету атестаційної роботи



Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Програмна реалізація

Arduino IDE

- проста у використанні;
- підсвічування коду;
- безкоштовна;
- має вбудований програматор;
- має багато корисних для розробки вбудованих функцій;
- оптимізована для роботи з платами Arduino.
- синтаксис мови візуально зрозумілий та читабельний;
- велика кількість програмних засобів та бібліотек;
- компілятор даної мови реалізований практично для всіх моделей мікроконтролерів;
- швидке виконання інструкцій.

Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Програмна реалізація

WinForm Microsoft .NET

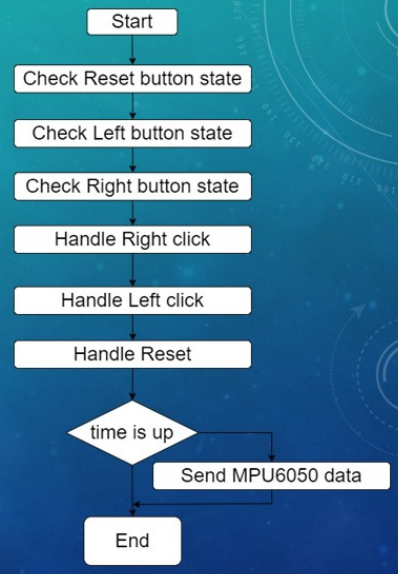
JetBrains Rider

- проста у використанні;
- IntelliSense;
- має студентськуу ліцензію;
- має вбудовані інструменту дебагінгу коду;
- має багато корисних для розробки вбудованих функцій;
- оптимізована для роботи з .Net платформою.
- синтаксис мови візуально зрозумілий та читабельний;
- велика кількість програмних засобів та nuget-пакетів;
- підтримка ООП;
- швидке виконання інструкцій;
- підтримує програмування для багатьох платформ.

Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

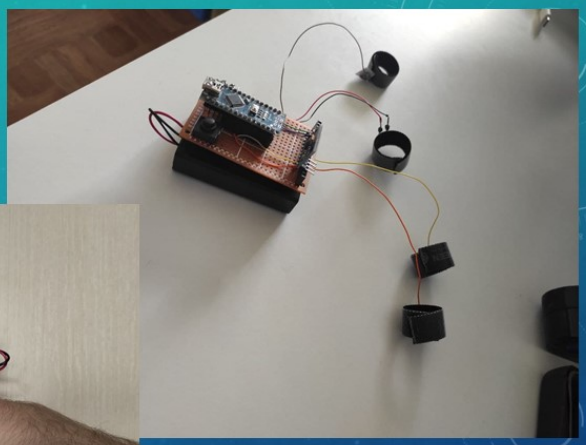
Алгоритм одного циклу роботи системи керування світлодіодами

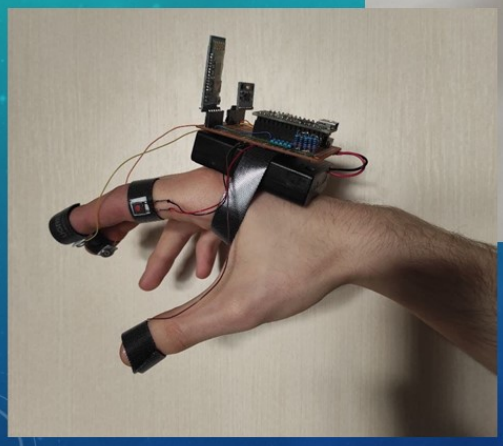
```
void loop() {
  B_Reset.tick();
  B_FingerLeft.tick();
  B_FingerRight.tick();

  RightClickHandler();
  LeftClickHandler();
  ResetButtonHandler();

  if(millis() - mpuTimer > 30){
    mpuTimer = millis();
    SendMPU6050Data();
  }
}
```



Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

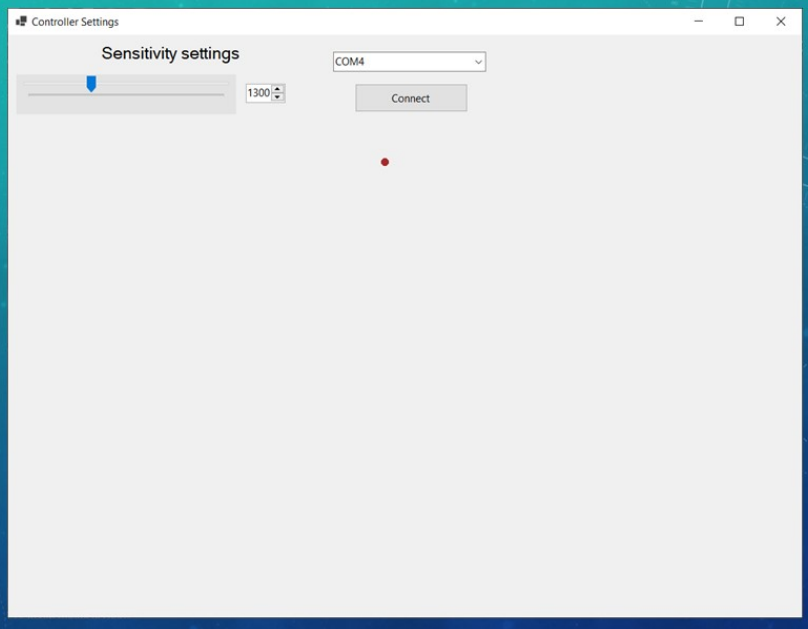
Фото розробленого пристрою





Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Фото розробленого комп’ютерного додатку



Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Переваги серед конкурентів

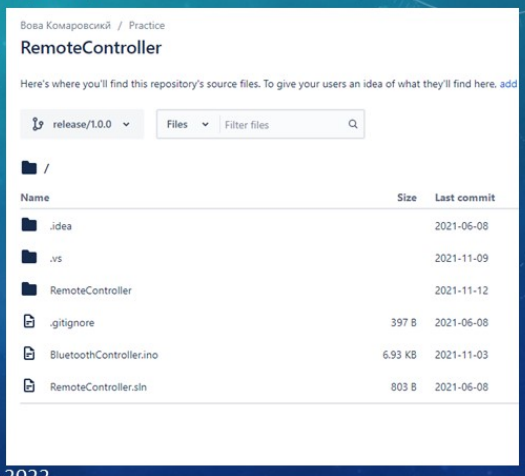
- відкритий програмний код;
- зручна архітектура для оновлення програмного забезпечення апаратної частини;
- архітектура оптимізована для заміни модулів та зручності ремонту;
- менша ціна;
- використана Bluetooth технологія.

Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Відкритий програмний код

Чому відкритий код це добре:
- часом користувачу просто хочеться мати можливість поглянути на код. Тобто не обов'язково навіть його мати, але щоб така можливість була. Це можуть бути параноїки безпеки у хорошому сенсі або просто борці за якісь права;

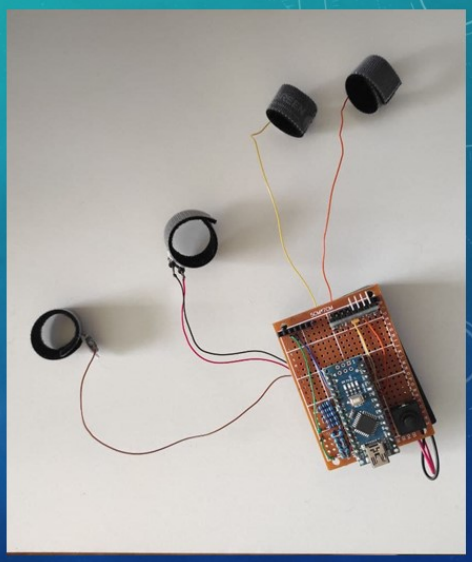
— користувач має можливість ввести редагування, навіть дуже серйозні;
— є можливість побудувати спільноту розробників, які разом працюють над вдосконаленням продукту.



Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Зручне оновлення програмного забезпечення апаратної частини

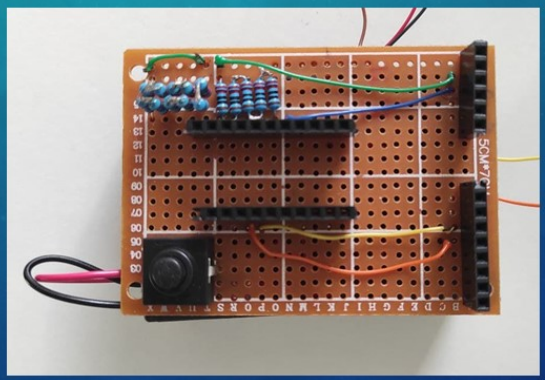
Маючи доступ до коду програмного забезпечення апаратної частини, необхідно мати можливість змінювати прошивку контролеру. Інакше не буде можливості вдосконалювати алгоритм роботи контролеру. Тому в розробленому проекті є відкритий доступ до Mini-USB порту Arduino Nano



Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

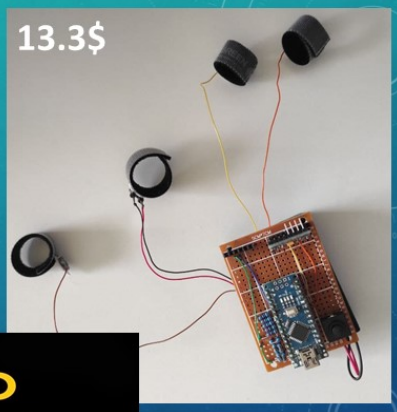
Архітектура оптимізована для заміни модулів та зручності ремонту

Апаратна частина приладу розроблена з використанням штиркового роз’єму PBS-40. Він дозволяє з легкістю замінювати головні модулі приладу (Arduino Nano, MPU-6050 та HC-05) без необхідності розпаювати плату.



Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

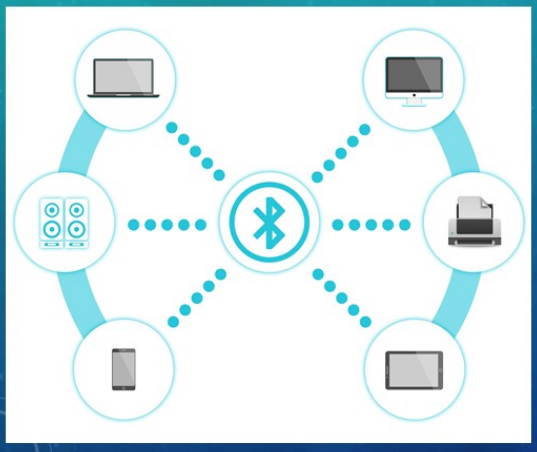
Менша ціна



Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Використана Bluetooth технологія

Найпопулярнішими технологіями бездротового зв’язку на цей час є Bluetooth та радіозв’язок. Головною перевагою Bluetooth є те, що він не потребує додаткового обладнання.



Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Висновки

У ході виконання атестаційної роботи було досягнуто поставлену мету – був створений контролер, який запропонував новий зручний інтерфейс роботи з обчислювальним пристроєм та який є значно дешевшим за конкурентів за рахунок використання модулю MPU-6050, Bluetooth модулю HC-05 та плати керування Arduino NANO.

Розглянуто властивості та особливості роботи з модулем MPU-6050 та HC-05. Розроблено структуру системи керування обчислювальним пристроєм, алгоритм керування, обрані відповідні програмні та апаратні компоненти системи. Створено макет системи бездротового керування персональним комп'ютером у вигляді контролеру, що керується платою Arduino Nano та кнопками.

Комаровський В.Е., ст. гр. СКСм-20-2, каф. АПОТ, 2022

Відомість кваліфікаційної роботи

№	Позначення	Найменування	Дод. відомості
		Текстові документи	
1	ГЮІК. 46741Х.001 ПЗ	Пояснювальна записка	86 с.
		Графічні документи	
2		Слайд-презентація	19 слайдів
		Носії інформації	
3		Файл пояснювальної записки	1 шт.

Змін.	Арк.	Номер докум.	Підп.	Дата	ГЮІК. 46741Х.001 ВД			
Розроб.		Комаровський В.Е.			Подієві HDL-моделі керуючих автоматів в системах логічного управління Відомість кваліфікаційної роботи	Літ	Аркуш	Аркушів
Перевір.		Шкіль О.С.				У		
Н.контр.		Шкіль О.С.				ХНУРЕ Кафедра АПОТ		
Затв.		Чумаченко С.В.						