МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра **САП**

ЗВІТ

до виконання розрахунково графічної роботи

На тему: ” розроблення цифрового компасу на МЕМС - магнітометрі”

з курсу “Мікропроцесорні системи ”

Виконав:

Студент гр.ПП-31

Гаврилюк Н. О.

Прийняв:

Доцент кафедри САП

Головатий А.І.

ЛЬВІВ - 2025

Мета: розробити та дослідити мікропроцесорний пристрій на базі Arduino Nano, який вимірює напрямок за допомогою аналогових сигналів та відображає його у вигляді ASCII-компасу. Передбачити проєктування апаратної частини, створення принципової схеми в Proteus, розроблення вбудованого програмного забезпечення та перевірку працездатності моделі під час симуляції.

1. **Використані компоненти та їх технічні характеристики**

У процесі виконання роботи як розробляв схему у програмному забезпечені Proteus, де немає компоненту HMC5883L, тому я використовував два потенціанометра для симуляції магнітного поля землі.

У процесі розроблення та моделювання цифрового компасу було використано такі апаратні компоненти та інструменти середовища симуляції:

1.1. Arduino Nano (ATmega328P)

Arduino Nano — це компактна плата на базі 8-бітного мікроконтролера ATmega328P, яка застосовувалась як основний обчислювальний модуль.

Технічні характеристики:

- Мікроконтролер: ATmega328P
- Робоча напруга: 5 В
- Живлення: 7–12 В через VIN або 5 В через USB
- Частота тактування: 16 МГц
- Flash-пам’ять: 32 КБ (з них 2 КБ зайняті бутлоадером)
- SRAM: 2 КБ
- EEPROM: 1 КБ
- Цифрові входи/виходи: 14 (6 із PWM)
- Аналогові входи: 8 (10-бітний АЦП)
- Тип USB-контролера: CH340G
- Комунікації: UART, I²C, SPI

У проекті плата виконувала такі функції:

- оцифрування сигналів із двох аналогових датчиків;
- обчислення кута напрямку за допомогою функції atan2();
- генерація ASCII-графічного зображення компаса;
- передавання інформації у Serial Monitor або Virtual Terminal в Proteus.

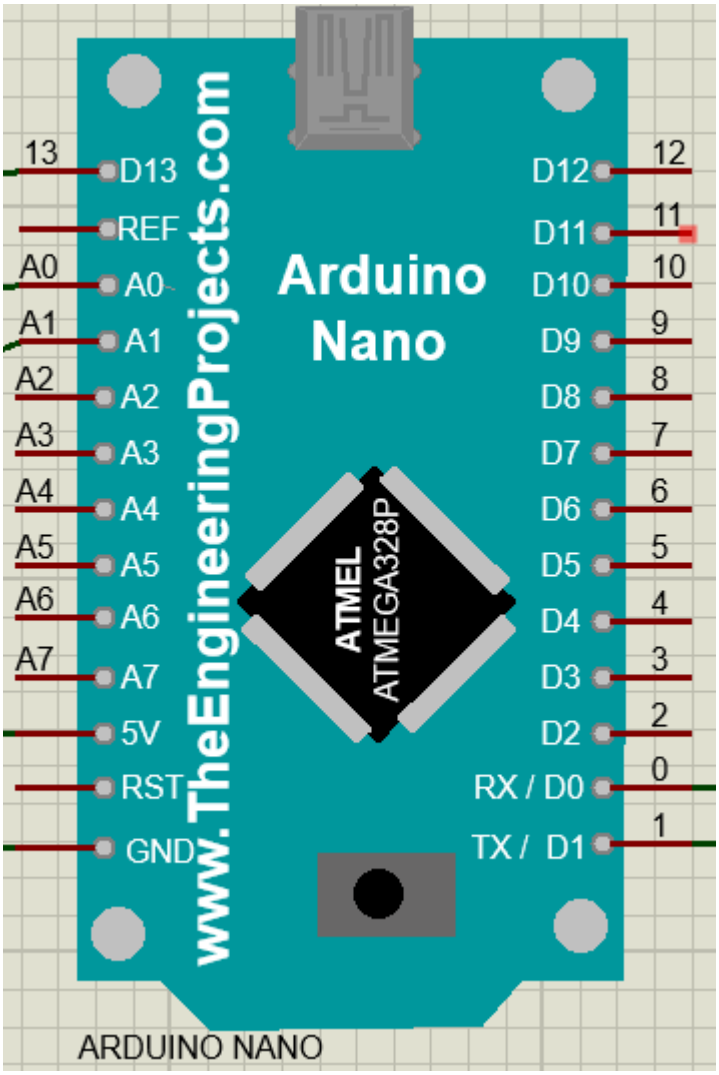

Рис. 1. Arduino Nano на схемі

Рис. 2. Arduino Nano

1.2. Потенціометри (2 шт.) — аналогові моделюючі елементи

Для симуляції зміни магнітного поля землі використовувалися два потенціометри, підключені до входів A0 та A1 мікроконтролера. У реальному пристрої їхню роль виконував би MEMC-магнітометр, але у моделі потенціометри дозволили імітувати аналогові координати X та Y.

Технічні характеристики стандартного потенціометра:

- Тип: обертовий, змінний опір
- Номінальний опір: 10 кОм
- Точність: ±20 %
- Вихідний сигнал: аналогова напруга 0…5 В
- Спосіб підключення: трьохвивідний (VCC — вихід — GND)

Призначення:

- Потенціометр №1 → імітація осі Y магнітометра
- Потенціометр №2 → імітація осі X магнітометра
- Формують два незалежні аналогові значення, що визначають кут напрямку.

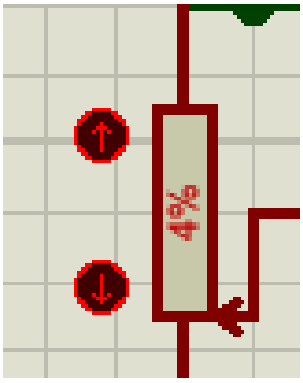

Рис. 3. Потенціанометр на схемі

Рис. 4. Потенціанометр

1.3. Serial Terminal (Virtual Terminal в Proteus)

Віртуальний термінал використовувався для відображення ASCII-компасу, сформованого Arduino Nano.

Характеристики:

- Робоча швидкість: 9600 бод
- Інтерфейс: UART (TX → RX терміналу)
- Можливість виводу текстової графіки
- Використовується лише для прийому даних

Призначення:

- відображення «графічного» компаса, побудованого символами ASCII;
- наочне тестування правильності роботи алгоритму у Proteus.

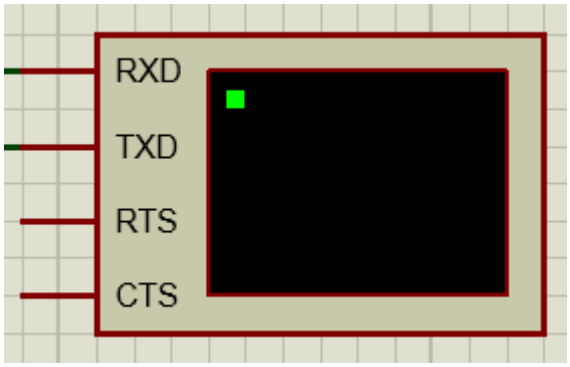

Рис. 5. Serial Terminal на схемі

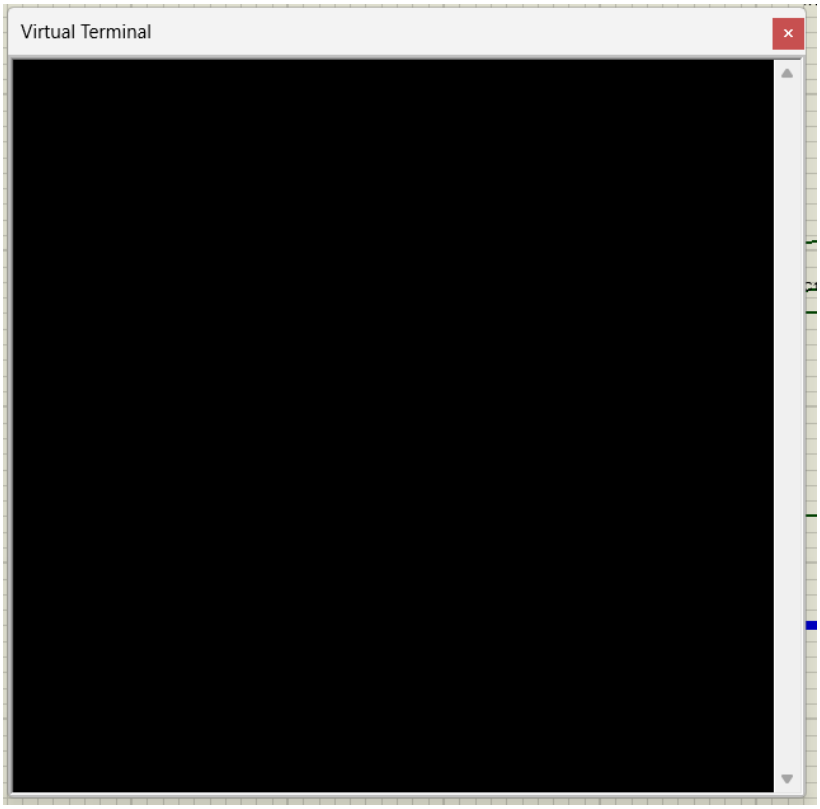

Рис 6. Virtual Terminal

1.4. Програмне забезпечення Proteus Design Suite

Для моделювання та створення принципової схеми застосовувалася середа Proteus 8 Professional.

Функціональні можливості, що були використані:

- Схемотехнічне моделювання Arduino Nano
- Підключення потенціометрів та UART
- Створення робочої схеми з живленням та АЦП
- Візуалізація роботи пристрою через Virtual Terminal
- Перевірка роботи програми у режимі симуляції

Proteus дозволяє завантажувати прошивку .hex, згенеровану Arduino IDE, що робить можливим повне відтворення роботи мікроконтролера без фізичних компонентів.

2. Опис програмного забезпечення

Для реалізації цифрового компаса було розроблено вбудоване програмне забезпечення мовою C/C++ у середовищі Arduino IDE. ПЗ виконує повний цикл обробки сигналів: від оцифрування даних АЦП до побудови ASCII-зображення.

2.1. Основні модулі програми

1. Ініціалізація мікроконтролера
    - налаштування аналогових входів A0 і A1;
    - запуск інтерфейсу UART зі швидкістю 9600 бод;
    - підготовка двовимірного масиву для ASCII-компасу.
2. Модуль зчитування аналогових сигналів
    - читання даних АЦП;
    - зсув відносно середини (512) для отримання діапазону −512…+511;
    - первинна фільтрація (проста перевірка на зміну).
3. Модуль обчислення кута
    - використання математичної функції atan2(y, x);
    - переведення радіан у градуси;
    - нормалізація результату до 0–360°;
    - конвертація в радіани для побудови стрілки.
4. Модуль побудови ASCII-компасу
    - очищення попередньої графіки;
    - побудова кола-компаса на основі рівняння кола $r^2 = x^2 + y^2$;
    - розрахунок координат стрілки;
    - відображення символів O, *, .;
    - додавання позначень напрямків N, S, W, E.
5. Модуль виводу в Serial Monitor
    - друк фінального ASCII-малюнка;
    - подвійний друк символів для симетрії;
    - нанесення центральної точки "C".

2.2. Алгоритм роботи пристрою

1. Ініціалізація схеми.
2. Зчитування сигналів з потенціометрів (імітація X та Y).
3. Обчислення кута повороту.
4. Нормалізація кута.

5. Побудова ASCII-компасу.
6. Передача результату у віртуальний термінал.
7. Оновлення лише при зміні значення (антидребезг 2 секунди).

2.3. Причини вибору Arduino IDE

- Доступність та підтримка ATmega328P
- Зручний інтерфейс компіляції та завантаження прошивки
- Вбудовані бібліотеки для роботи з UART та математики (atan2)
- Простота редагування та налагодження коду
- Сумісність із Proteus (генерація .hex файлу)

Схема підключень

Arduino	**Підключення**	**Опис**
RX	**TXD Serial Monitor**	**Прийом даних у Serial Monitor**
TX	**RXD Serial Monitor**	**Передача даних у Serial Monitor**
A0	**Потенціометр X**	**Зчитування осі X**
A1	**Потенціометр Y**	**Зчитування осі Y**
5V	**Живлення**	**Подача живлення на модулі**
GND	**GND**	**Спільна земля**

Код написаний у Arduino IDE:

```
char arr[29][29];


int arrow_len = 10; // довжина стрілки
int cx = 14; // центр x
int cy = 14; // центр y
int r = 13;  // радіус
float theta; // кут у градусах
```

```
float rad; // кут у радіанах
int currentTime,lastTime=-50; // дребезг
int x_adc=0, y_adc=0;// зчитування напруги

float readCompassAngle() {
  x_adc = analogRead(A0) - 512;
  y_adc = analogRead(A1) - 512;

  float x = (float)x_adc;
  float y = (float)y_adc;

  float angle_rad = atan2(y, x);

  float angle_deg = angle_rad * 180.0 / 3.14159;

  if (angle_deg < 0) angle_deg += 360.0;

  return angle_deg;
}

void showCompas()
{
  for(int i=0;i<29;i++)
  {
    for(int j=0;j<29;j++)
    {
      if (arr[i][j]=='*')
      {
```

```
      arr[i][j]='.';
    }
  }
}

  for (int k = 1; k <= arrow_len; k++) {
int x = cx + k * cos(rad);
int y = cy + k * sin(rad);
arr[y][x] = '*';
}
arr[14][14]='C';

for(int i =0;i<cx*2;i++)
{
  Serial.print(" ");
}
Serial.print("N");
Serial.println();
for (int i = 0; i < 29; i++) {
  for (int j = 0; j < 29; j++) {
    if(i==cx)
    {
      if(j==0)
      {
        Serial.print("W");
        Serial.print(arr[i][j]);
        continue;
      }
```

```
            if(j==cx*2)
        {
          Serial.print(arr[i][j]);
          Serial.print("E");
          continue;
        }
      }
      Serial.print(arr[i][j]);
      Serial.print(arr[i][j]);
    }
    Serial.println();
  }
    for(int i =0;i<cx*2;i++)
  {
    Serial.print(" ");
  }
  Serial.print("S");
  Serial.println();
}

void setup() {
  pinMode(A0, INPUT_PULLUP);
  pinMode(A1, INPUT_PULLUP);
  Serial.begin(9600);

  for (int i = 0; i < 29; i++) {
    for (int j = 0; j < 29; j++) {
```

```
      int dx = j - cx;
      int dy = i - cy;
      int dist2 = dx*dx + dy*dy;

      if (dist2 >= (r-1)*(r-1) && dist2 <= (r+1)*(r+1)) {
        arr[i][j] = 'O';
      }
      else if (dist2 < (r-1)*(r-1)) {
        arr[i][j] = '.';
      }
      else {
        arr[i][j] = ' ';
      }
    }
  }
}

void loop() {
  currentTime=millis();
  if((x_adc != analogRead(A0) - 512 || y_adc != analogRead(A1) - 512) &&
currentTime-lastTime>=2000)
  {
    theta = readCompassAngle();
    theta = 360 - theta;
    if (theta >= 360) theta -= 360;
    rad = theta * 3.14159 / 180.0;
    showCompas();
    lastTime=currentTime;
    x_adc = analogRead(A0)- 512;
```

```
    y_adc = analogRead(A1)- 512;
  }
}
```

3. Дослідження моделі МП-пристрою в Proteus

У Proteus виконано моделювання роботи пристрою. На схемі Arduino Nano підключено до віртуального термінала, що дозволяє бачити ASCII-компас у режимі симуляції. При зміні значень потенціометрів (імітація датчика) кут змінюється, а стрілка на віртуальному компасі повертається у відповідному напрямку. Схеми, графіки та діаграми показують стабільність роботи АЦП, коректне перетворення напруги в кут, та правильне відтворення зображення у терміналі. Модель повністю повторює роботу реального пристрою на Arduino Nano.

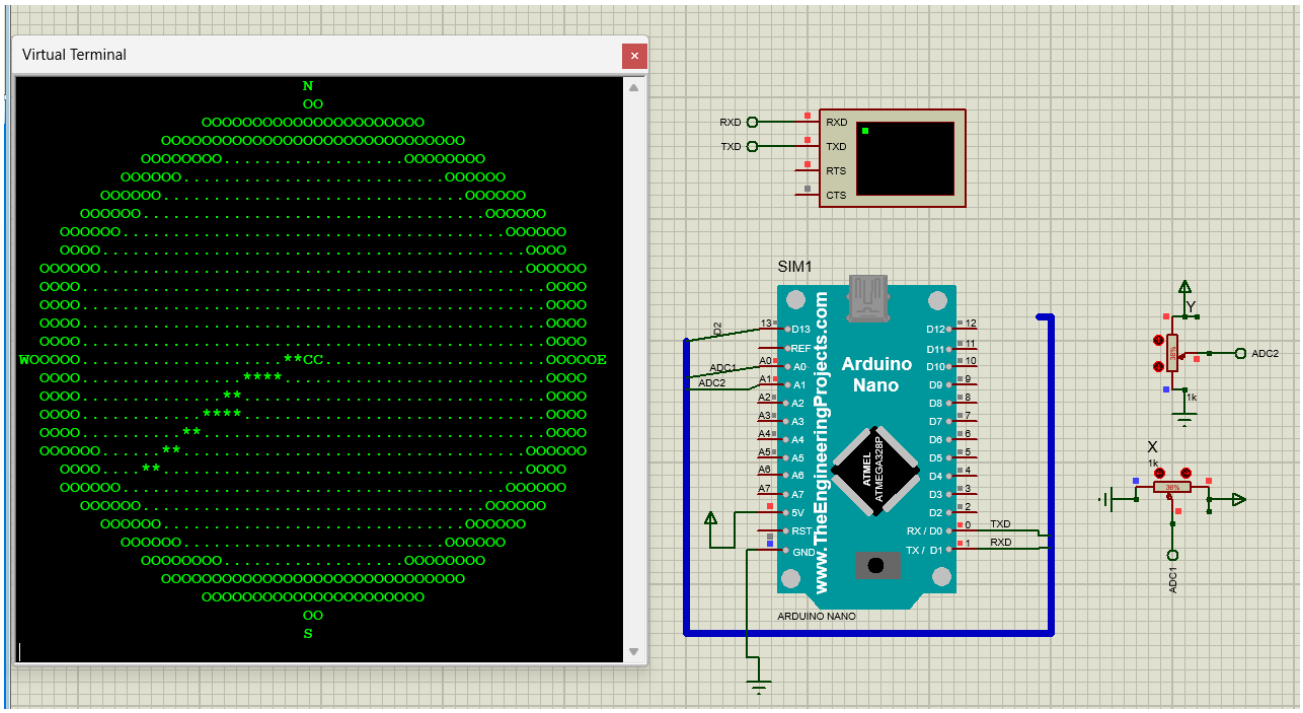

Рис. 7. Імітація роботи компасу у virtual terminal

Висновок: у результаті виконання роботи було розроблено та змодельовано мікропроцесорний пристрій на базі Arduino Nano, здатний вимірювати напрямок і відображати його у вигляді ASCII-компасу. Створено принципову схему, блок-схему та вбудоване ПЗ. Проведене моделювання у Proteus підтвердило правильність роботи алгоритму та апаратної частини. Проєкт може бути розширений шляхом додавання реального магнітного датчика (HMC5883L), OLED-екрана або бездротової передачі даних.