ЛАБОРАТОРНА РОБОТА №6

Дослідження роботи GPS-модуля за допомогою Arduino UNO та LCD-дисплею

1.1 Мета роботи

Розробити код програми для Arduino з використанням NEO-6M GPS-модуля з метою визначення часу та географічних координат місцезнаходження, виведення отриманих даних на LCD-дисплей.

1.2. Короткі теоретичні відомості

Модуль GPS NEO-6M GY-GPS 6MV2 NEO6MV2 можна використовувати разом з ARDUINO та іншими мікроконтролерними системами. Обмін даними здійснюється через UART. Модуль має вбудовану пам'ять EEPROM для збереження даних після вимкнення живлення.

Рисунок 1.1 – GPS модуль

Обмін даними відбувається через UART на швидкості від 4800 до 230400 бод, за умовчанням встановлена швидкість 9600. Чіп здатний відстежувати до 22 супутників одночасно на 50 каналах з більшим рівнем чутливості до -181 дБ. Споживання струму становить лише 45 мА. Рівень живлення чіпа 2,7-3,6, яке забезпечується стабілізатором напруги MIC5205 з вихідним рівнем 3.3 В.

До модуля додається активна керамічна антена, яку слід попередньо підключити до відповідного роз'єму. Дуже простий у використанні. Вивід CS використовується для вибору інтерфейсу. Якщо на вході CS низьких рівень - використовується SPI інтерфейс. Якщо ж на вході CS високий рівень - використовується I2C інтерфейс.

Характеристики:

— Живлення: 3.3 - 5 В;

— Модуль U-blox NEO-6M GPS;

— U-blox 6 має 50 каналів позиціонування з більш ніж 2 мільйонами ефективних кореляторів;

— Вбудована батарея для швидкого, холодного старту;

— Швидкість оновлення розташування 5Hz;

— Оснащений активною керамічною антеною;

— EEPROM (пам'ять) для зберігання налаштувань;

— Підтримка SBAS (WAAS, EGNOS, MSAS, GAGAN);

— Акумулятор для резервного збереження інформації;

— Інтерфейс UART TTL.

На модулі розташований єдиний світлодіод, котрий сигналізує про стан роботи: швидке миготіння – пошук супутників та збір інформації, повільне – дані отримані та передаються до порту.

Для роботи модуля достатньо подати на нього живлення та хрестоподібно з'єднати роз'єми RT-TX модуля та контролера. У нашому прикладі як контролер ми будемо використовувати Arduino UNO. UNO має лише один апаратний UART на пінах 0 і 1, який нам знадобиться для зв'язку з комп'ютером, тому для підключення GPS-модуля використовуємо програмний UART на пінах 3 (TX) і 4 (RX). Зрозуміло, за потреби можна призначити будь-які інші піни.

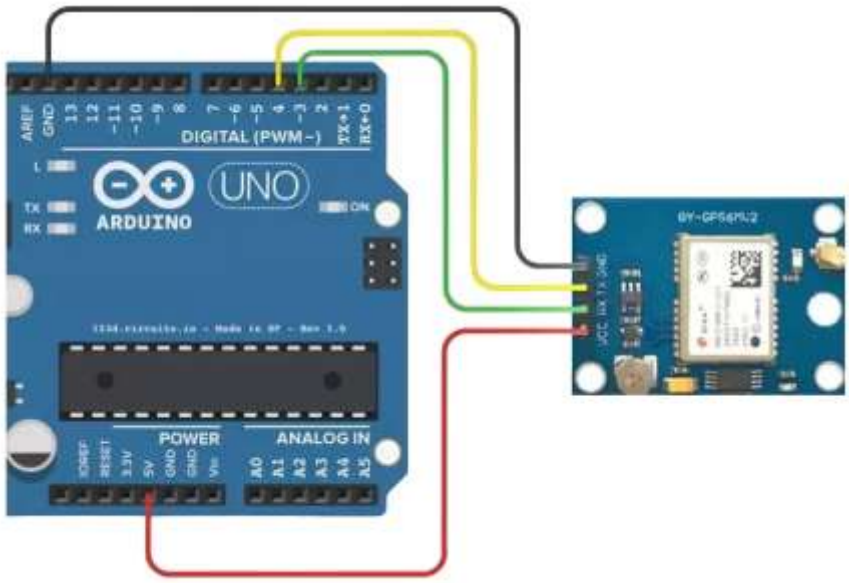

Рисунок 1.2 – Схема підключення GPS-датчика до Ардуіно

Після підключення, доведеться перевести інформацію отриману від модуля з мови NMEA на географічні координати. Для цього є два способи: складний та легкий.

```
$GPGGA
,110617.00,41XX.XXXXX,N,00831.54761,W,1,05,2,68,129.0,M,50.1,M,,*42
```

Перший, складний – парсинг рядків. Під цим терміном мається на увазі розбір рядків на складові елементи, спираючись на їх вміст та порядок розташування. Знаходимо поєднання букв “GPGGA”, отже перед нами рядок з основними даними, розбиваємо їх у шматки цифр і символів, затиснутих між комами, нумеруємо, аналізуємо вміст. 110617 - час UTC 11.06.17, 41XX.XXXX,N - 41 градус і стільки хвилин північної широти, потім довгота, якість фіксації, кількість супутників, тощо. І так по кожному рядку. Зрозуміло, слід заздалегідь підготуватися і зібрати повну інформацію про всі рядки, їх заголовки і порядок розташування даних.

Другий, легкий – використовувати бібліотеки, де парсинг уже зроблений. Бібліотек досить багато, тільки в самій IDE Arduino знайшлося більше десятка.

Скетч для обробки даних від GPS-модуля:

```
// ПРИКЛАД ВИВОДИТЬ ДАНІ NMEA ОТРАВНІ ПО UART: // дата, час, координати, швидкість, напрямок.
//
const uint8_t pinRX = 6;                                  //  визначаємо висновок RX (програмного UART) на платі Arduino до якого підключено виведення TX модуля. Можна змінити номер виводу.
const uint8_t pinTX = 7;                                 //  Визначаємо виведення TX (програмного UART) на платі Arduino, до якого підключено виведення RX модуля. Можна змінити номер виводу.
//
#include   <iarduino_GPS_NMEA.h>                         //  Підключаємо бібліотеку для розшифрування рядків протоколу NMEA, що одержуються за UART.
#include    <SoftwareSerial.h>          //  Підключаємо бібліотеку для роботи із програмним UART.
//
iarduino_GPS_NMEA gps;                                   //  Оголошуємо об'єкт gps для роботи з функціями та методами бібліотеки iarduino_GPS_NMEA.
SoftwareSerial    SerialGPS(pinRX, pinTX);         //  Объявляем объект SerialGPS для работы с функциями и методами библиотеки SoftwareSerial, указав выводы RX и TX Arduino.
//
char* wd[] = {"Вс", "Пн", "Вт", "Ср", "Чт", "Пт", "Сб"}; //  Визначаємо масив рядків, що містять по дві перші літери з назв дня тижня.
//
void setup() {                                           //
  Serial.begin(9600);                                  //  Ініціюємо роботу з апаратною шиною UART для виведення даних у монітор послідовного порту на швидкості 9600 біт/сек.
  SerialGPS.begin(9600);                                   //   Ініціюємо роботу із програмною шиною UART для отримання даних від GPS модуля на швидкості 9600 біт/сек.
  gps.begin(SerialGPS);                                // Ініціюємо розшифрування рядків NMEA вказавши об'єкт використовуваної шини UART (замість програмної шини, можна вказувати апаратні: Serial, Serial1, Serial2, Serial3).
  gps.timeZone(3);                                     //  Вказуємо часовий пояс (±12 годин) або GPS_AutoDetectZone для автоматичного визначення часового поясу за довготою.
}                                                        //
//
void loop() {                                            //
```

```
  gps.read();                                          // Читаємо дані (читання може тривати більше 1 секунди). Функції можна вказати масив для отримання даних про супутники.
  /* Час:                                               */  Serial.print(gps.Hours); Serial.print(":");         Serial.print(gps.minutes);         Serial.print(":"); Serial.print(gps.seconds); Serial.print(" ");
  /* Дата:                                              */  Serial.print(gps.day  ); Serial.print(".");     Serial.print(gps.month          );     Serial.print("."); Serial.print(gps.year   ); Serial.print("г (");
  /* Додаткові дані дати:                            */   Serial.print(wd[gps.weekday]); Serial.print("), UnixTime: "); Serial.print(gps.Unix); Serial.print(". ");
  /*  Координати (широта, довгота, висота):          */    Serial.print("Ш:  "); Serial.print(gps.latitude,       5);       Serial.print("°,       Д:       "); Serial.print(gps.longitude,       5);       Serial.print("°,       В:       "); Serial.print(gps.altitude, 1); Serial.print("м. ");
  /*Рух (швидкість, курс):                          */   Serial.print("Швидкість: "); Serial.print(gps.speed);   Serial.print("км/ч,   ");   Serial.print(gps.course); Serial.print("°. ");
  /* Супутники (активні/спостерігаються):           */  Serial.print("Супутники: ");         Serial.print(gps.satellites[GPS_ACTIVE]);         Serial.print("/"); Serial.print(gps.satellites[GPS_VISIBLE]); Serial.print(". ");
  /*Геометричний фактор погіршення точності:  */    Serial.print("PDOP:  "); Serial.print(gps.PDOP);   Serial.print(",   HDOP:   ");   Serial.print(gps.HDOP); Serial.print(", VDOP: "); Serial.print(gps.VDOP); Serial.print(". ");
  /* Помилка визначення часу:                */  if (gps.errTim) {
    Serial.print("Час недостовірний. "           );
  }
  /* Помилка визначення дати:                */  if (gps.errDat) {
    Serial.print("Дата недостовірна. "           );
  }
  /*Помилка позиціонування:                 */  if (gps.errPos) {
    Serial.print("Координати недостовірні. "     );
  }
  /*Помилка визначення швидкості та курсу */  if (gps.errCrs) {
    Serial.print("Швидкість та курс недостовірні. ");
  }
  /*                                              */  Serial.print("\r\n");
  delay(5000); // період опитування, без затримки видаватиме приблизно 1 раз на секунду}
```

1.3. Підготовка до роботи

При підготовці до роботи необхідно:

- ознайомитись з рекомендованою літературою;
- вивчити короткі теоретичні відомості.

1.4. Порядок роботи:

1. Підключити модуль ArduinoUno до GPS-модуля.
2. Скачати та підключити в скетчі необхідні бібліотеки для роботи з GPS-модулем NEO-6M та LCD-дисплеєм.
3. Запустити код програми з теорії лабораторної роботи для виводу даних на монітор порта.
4. Зібрати макет з ArduinoUno, GPS-модуля та LCD дисплею.
5. Скорегувати код програми для виведення географічних координат, часу та швидкості на LCD дисплей.
6. Зробити 3-5 скріншотів робочого макету.
7. Знову повернутися до коду програми для виводу даних на монітор порта.
8. Обережно рухати датчик GPS-модуля для зміни швидкості.

9. Побудувати графіки залежності зміни координат від часу.

10. Побудувати графіки залежності зміни швидкості від часу.

11. Отримані координати з GPS-модуля необхідно порівняти з координатами google maps та оцінити точність.

12. Оформити звіт та зробити висновки.