Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Факультет прикладной математики и механики

Кафедра «Прикладной математики»

Направление: 24.03.02 «Системы управления движением и навигация»

(уровень бакалавриата)

Отчет о проделанной работе

**По дисциплине:** Системное программировние

**По теме:** Способы беспроводной передачи данных между Arduino и ПК

Выполнил студент гр. ИВК-21-1б

Каримов Р.А.

*(Ф.И.О.)*

Проверил:

Старший преподаватель

кафедры ПМ Абакшин Д.С.

*(должность, Ф.И.О.)*

**Пермь 2025**

**1. Беспроводная передача данных с помощью Bluetooth**

**Описание:** использует модули Bluetooth, такие как HC-05 или HC-06, которые подключаются к Arduino. ПК подключается к модулю как к стандартному Bluetooth-устройству. Данные передаются по стандартному протоколу Bluetooth.

**Преимущества:** Низкая стоимость модулей, простота использования (многие библиотеки доступны для Arduino и ПК), низкое энергопотребление.

**Недостатки:** относительно низкая скорость передачи данных, ограниченная дальность действия (обычно до 10 метров), необходимость сопряжения устройств.

**Необходимое оборудование:**

- Arduino (любая модель): Uno, Nano, Mega;

- Bluetooth модуль HC-05 или HC-06: эти модули недорогие и широко распространены;

соединительные провода: для подключения Bluetooth модуля к Arduino;

ПК с Bluetooth: большинство современных ПК имеют встроенный Bluetooth;

USB кабель: для прошивки кода на Arduino.

**Подключение Bluetooth модуля к Arduino:**

HC-05 / HC-06 имеет 4 или 6 выводов (в зависимости от модели). Нас интересуют:

VCC: подключается к 5V на Arduino;

GND: подключается к GND (земле) на Arduino;

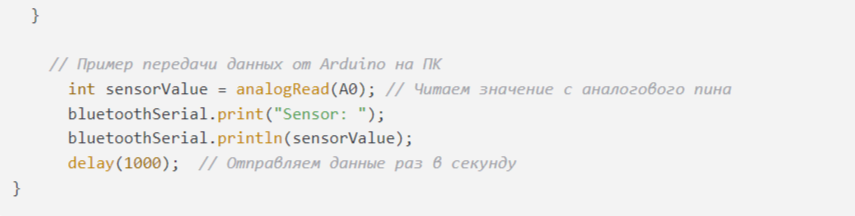
TXD: подключается к пину RX на Arduino (обычно цифровой пин, нужно использовать программный Serial);

RXD: подключается к пину TX на Arduino (обычно цифровой пин, нужно использовать программный Serial);

Важно: TXD модуля подключается к RX Arduino, а RXD модуля - к TX Arduino (перекрестное соединение).

**Arduino код:**

****

****

**Пояснение Arduino кода:**

1. #include <SoftwareSerial.h>: подключаем библиотеку для создания программного серийного порта (т.к. аппаратный серийный порт используется для связи с компьютером по USB).

2. const int rxPin = 10, const int txPin = 11: Определяем пины, к которым подключены TXD и RXD Bluetooth модуля. Вы можете использовать другие цифровые пины, если необходимо.

3. SoftwareSerial bluetoothSerial(rxPin, txPin); создаем объект bluetoothSerial для работы с программным серийным портом.

4. bluetoothSerial.begin(9600); инициализируем скорость передачи данных для связи с Bluetooth модулем (обычно 9600).

5. if (bluetoothSerial.available()): проверяем, пришли ли данные через Bluetooth.

6. char data = bluetoothSerial.read();: Читаем принятые данные.

7. bluetoothSerial.print("Sensor: "); и bluetoothSerial.println(sensorValue);: Отправляем строку “Sensor:” и значение с аналогового пина A0 на ПК через Bluetooth.

8. if (data == '1') { ...} else if (data == '0') { ... }: Пример управления цифровым пином 13 Arduino по командам “1” и “0” отправленных с ПК.

**Код на Python для ПК:**

****

**Пояснение Python кода:**

1. import serial: импортируем библиотеку для работы с серийным портом.

2. bluetooth\_port = 'COM4': замените 'COM4' на COM-порт, который назначен Bluetooth модулю на ПК. Можно посмотреть в диспетчере устройств Windows.

3. ser = serial.Serial(bluetooth\_port, 9600): создаем объект serial.Serial для связи с Bluetooth модулем на ПК.

4. ser.in\_waiting: проверяет, есть ли данные на порту

5. line = ser.readline().decode('utf-8').rstrip(): читаем данные с порта, декодируем их в строку и удаляем лишние пробелы и символы конца строки.

6. command = input(...): ждем ввода команды от пользователя (1 или 0).

7. ser.write(command.encode()): отправляем команду на Arduino.

**Как использовать:**

1. Загружаем Arduino код на плату.

2. Устанавливаем Bluetooth сопряжение между ПК и Bluetooth модулем.

3. Запускаем Python скрипт.

4. Смотрим в терминале Python, должны выводиться данные с датчика Arduino, а при вводе команды “1” или “0” будет включаться или отключаться светодиод на 13 пине Arduino

5. Можно отправлять сообщения через Python скрипт (например, 1 для включения светодиода на пине 13 Arduino, 0 - для выключения)

**2. Беспроводная передача данных с помощью Wi-Fi**

**Описание:** использует модули Wi-Fi, такие как ESP8266 или ESP32, которые имеют встроенный процессор и Wi-Fi. Arduino может напрямую подключаться к сети Wi-Fi и передавать данные на сервер, к которому подключается ПК. Или может использовать протокол UDP/TCP для прямой связи.

**Преимущества:** большая дальность действия, высокая скорость передачи данных, возможность подключения к интернету.

**Недостатки:** более высокая стоимость модулей, более сложное программирование, потребляет больше энергии, чем Bluetooth.

**Необходимое оборудование:**

1. Arduino с ESP8266 или ESP32: ESP8266 — более дешевый, но ESP32 мощнее и функциональнее;

2. ПК с сетевым подключением: для запуска серверной части;

3. USB кабель: для прошивки кода на ESP32/ESP8266.

**Выбор протокола:** для связи чаще всего используются:

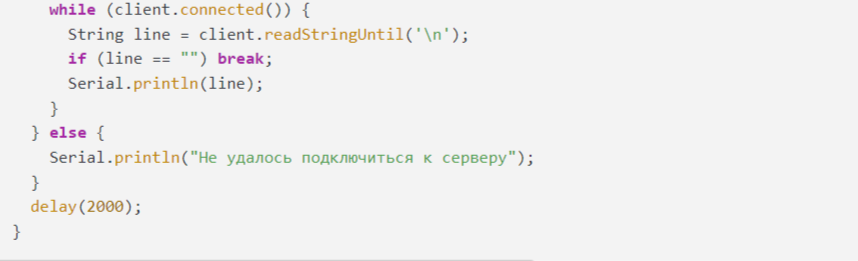
- HTTP: простой протокол, но не самый эффективный для постоянного потока данных;

- MQTT: более эффективный протокол для обмена сообщениями “публикация-подписка”, идеально подходит для IoT-устройств. Требуется MQTT брокер;

- TCP/UDP: более низкоуровневые протоколы, дают больше контроля, но требуют более сложной реализации.

**Пример с использованием HTTP и Python:**

****

****

**Python код (сервер на ПК):**



**Объяснение:**

- Объяснение: ESP32 код: подключается к Wi-Fi, затем отправляет HTTP GET запрос на сервер, включая значение датчика в запросе;

- Python код: создает простой HTTP сервер, который обрабатывает GET запросы и выводит значение датчика в консоль.

**3. Беспроводная передача данных с помощью nRF24L01 (Радиомодуль)**

**Описание:** использует радиомодули nRF24L01 для беспроводной связи. Для связи на ПК понадобится соответствующий USB-адаптер.

**Использование:** требуется библиотека для работы с nRF24L01 на Arduino. Два модуля (на Arduino и на ПК-адаптере) настраиваются на одном частотном канале. Arduino передает данные, а адаптер на ПК их принимает. Код сложнее, чем для последовательного порта, и требует настройки адресов.

**Необходимое оборудование:**

1. Arduino (любая модель): Uno, Nano, Mega и т.д.

2. Два модуля nRF24L01+: Один для Arduino, а второй для подключения к ПК.

3. USB-адаптер для nRF24L01: Этот адаптер подключается к ПК и позволяет общаться с модулем. (Можно и другой Arduino использовать как адаптер)

4. Соединительные провода: Для подключения nRF24L01 к Arduino и адаптеру.

**Подключение nRF24L01 к Arduino:**

**- nRF24L01 имеет следующие выводы:**

* GND: подключается к GND Arduino
* VCC: подключается к 3.3V (!!!) Arduino. 5V может повредить модуль.
* CE: подключается к цифровому пину Arduino (например, 9)
* CSN: подключается к цифровому пину Arduino (например, 10)
* SCK: подключается к пину SCK Arduino (обычно 13)
* MOSI: подключается к пину MOSI Arduino (обычно 11)
* MISO: подключается к пину MISO Arduino (обычно 12)

**- Подключение nRF24L01 к ПК через USB-адаптер:**

* Подключение может зависеть от вашего адаптера. В большинстве случаев нужно будет подключить GND, VCC (3.3V), CE, CSN, SCK, MOSI, MISO к соответствующим выводам адаптера.

**Arduino код:**

****

**Объяснение Arduino кода:**

* #include <SPI.h>, #include <nRF24L01.h>, #include <RF24.h>: подключаем библиотеки для работы с SPI и nRF24L01.
* #define CE\_PIN 9, #define CSN\_PIN 10: Определяем пины для CE и CSN.
* const byte address[6] = "00001";: устанавливаем адрес для радиосвязи (должен быть одинаковым на обоих модулях).
* RF24 radio(CE\_PIN, CSN\_PIN);: Создаем объект для управления модулем.
* radio.begin();: инициализируем радиомодуль.
* radio.openWritingPipe(address);: устанавливаем адрес для передачи.
* radio.stopListening();: готовимся к передаче (для передачи данных с Arduino).
* radio.write(&data, sizeof(data));: отправляем данные (значение датчика).

**Код для ПК (на Python) - С использованием адаптера:**

1. Установите драйвер и библиотеку для вашего USB-адаптера nRF24L01.

2. Узнайте COM-порт вашего USB-адаптера в диспетчере устройств.

3. Предположим, что после настройки адаптера, вы можете получать данные через серийный порт, тогда код для ПК будет выглядеть так:



**Объяснение Python кода:**

* import serial: импортирует библиотеку для серийной связи.
* COM\_PORT = 'COM5': замените COM5 на фактический COM-порт вашего USB-адаптера nRF24L01.
* BAUD\_RATE = 115200: установите скорость передачи данных, аналогичную Serial.begin() в коде Arduino.
* try...except: используется для обработки ошибок при подключении к COM порту и при чтении данных.
* ser.in\_waiting: проверяет есть ли данные на порту
* line = ser.readline().decode('utf-8').rstrip(): Читает данные с порта и декодирует их.

**Как использовать:**

1. Загрузите Arduino код на плату.

2. Подключите nRF24L01 к Arduino и к USB-адаптеру.

Запустите Python скрипт на ПК.

3. Вы должны видеть значения с датчика, поступающие на ПК из терминала Python.

**Важные моменты:**

1. Питание 3.3V: nRF24L01 работает от 3.3V.

2. Адрес: адреса для передачи и приема должны совпадать на обоих устройствах.

3. Библиотеки: убедитесь, что установлены библиотеки SPI, nRF24L01, RF24 в Arduino IDE.

4. Драйверы: установите драйверы для USB-адаптера nRF24L01.

5. Дальность: дальность nRF24L01 зависит от условий и антенны.

6. Настройка адаптера: Инструкции по настройке адаптера могут отличаться в зависимости от модели. Убедитесь, что все настроено верно.

**Дополнительные возможности:**

1. Можно передавать не только строки, но и другие типы данных (массивы байтов).

2. Можно реализовать двунаправленную связь.

3. Можно создавать сети с несколькими устройствами.

4. nRF24L01 требует более тщательной настройки и обработки ошибок, чем Bluetooth, но он дает больше контроля над радиосвязью.

**Заключение**

Выбор метода зависит от конкретных задач:

Дальность связи: Wi-Fi предлагает наибольшую дальность.

Скорость передачи данных: Wi-Fi самые быстрые.

Стоимость: Bluetooth и nRF24L01 — самые дешевые варианты.

Сложность программирования: Bluetooth относительно прост в использовании, а Wi-Fi более сложен.

Энергопотребление: Bluetooth и nRF24L01 — энергоэффективные решения.