Московский государственный университет
им. М. В. Ломоносова
Физический Факультет

Arduino и LabView: разработка метеостанции и программного обеспечения для сбора данных о погоде

Проект по основам автоматизации эксперимента.

Гадиров Эльвин.
Преподаватель: В. А. Соколов.

Баку 2023

Аннотация

Данная работа посвящена разработке метеостанции с использованием платформы Arduino, созданию mesh-сети из приемо-передатчиков и написанию программы на LabView для анализа полученных данных.

В работе представлен обзор плат Arduino и ее возможностей для разработки метеостанции. Исследованы различные датчики, используемые для измерения погодных параметров, таких как температура, влажность, атмосферное давление.

Для создания беспроводной сети передачи данных между метеостанцией и базовой станцией была использована mesh-технология. Рассмотрены принципы работы mesh-сети, настроены передатчики для обеспечения стабильной связи.

Особое внимание уделено программированию на LabView. Разработана программа, позволяющая считывать данные с метеостанции, анализировать их и визуализировать в удобном виде.

Полученные результаты показывают эффективность и надежность разработанной метеостанции, а также функциональность программы на LabView для обработки и представления данных. Работа может быть использована в качестве основы для дальнейших исследований в области метеорологии и разработки систем мониторинга погоды.

Оглавление

Листинги

Исходный код и остальные файлы:
`https://github.com/kelv1n9/Weather_Station`

Глава 1

Обзор технологий

1.1 Платформа Arduino

В качестве вычислительных устройств были использованы платы Arduino Nano. Arduino представляет собой платформу для разработки электронных проектов. Это небольшая плата с микроконтроллером ATmega328 и памятью, на которой присутствуют контакты ввода/вывода данных.

К таким платам можно подключать внешние датчики или другие платы (экран или приемник) и написав програмное обеспечение, автоматизировать некоторые процессы. Arduino имеет удобную среду разработки (Arduino IDE), которая основана на языке программирования $C/C++$.

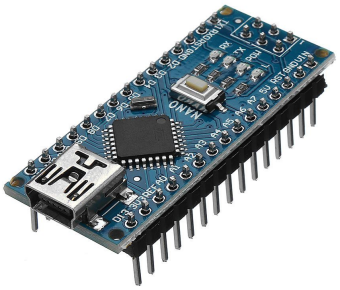

Рис. 1.1: Плата Arduino Nano

В этом проекте мы будем использовать три платы Arduino Nano: одна из них будет приемной (RX), а две остальные - передающими (TX[1]).

1.2 Датчики и модуль связи

В данной работе используются:

1. nRF24L01+ - Модуль беспроводной связи с усилителем мощности, платой питания и внешней антенной.
   (a) Напряжение питания: 3,3 В;
   (b) Скорость передачи данных: 250 – 2000 Кбит/сек;
   (c) Дальность связи при прямой видимости (с УМ): до 2000 метров.
2. DS18B20 - Герметичный датчик температуры.
   (a) Диапазон измеряемых температур: $-55°C... + 125°C$;

[1]TX - transmit - передача данных, RX - receive - получение данных

(b) Напряжение питания: 3...5,5 В;

(c) Интерфейс: 1-Wire.

3. DH21 - Датчик температуры и влажности.

   (a) Диапазон измеряемой температуры: $-40°C... + 80°C$;

   (b) Диапазон влажности: $0 - 99,9\%$;

   (c) Напряжение питания: 3,3...5 В.

4. BMP180 - Датчик атмосферного давления и температуры.

   (a) Диапазон измеряемой температуры: $-45°C... + 85°C$;

   (b) Диапазон измерения давления: 30...110;

   (c) Напряжение питания: 1.8...3.6 В.

5. Фоторезистор в качестве датчика освещенности.

Глава 2

Проектирование метеостанции

2.1 Требования к функциональности

Метеостанция должна быть способна собирать данные о различных погодных параметрах, таких как температура, влажность, атмосферное давление и иметь возможность отправлять их на базовую станцию. После принятия данных, программа, написанная на LabView, должна визуализировать принятые данные, показывая изменение параметров за, например, определенные промежутки времени.

Базовая станция должна работать как с одним из передатчиков, так и с несколькими (в нашем случае двумя) передающими устройствами. Для этого будет написан с нуля специальный алгоритм mesh-сети для плат Arduino.

2.2 Подключение периферии к Arduino

2.2.1 Устройство 1

Первое устройство имеет на борту датчики BMP180 и фоторезистор, а также модуль nRF24l01. Для работы прибора была написана следующая программа: подключенные платы инициализируются путем подключения библиотек и вызова соответствующих функций. Также объявляем пины микроконтроллера, которым будем подключать датчики и связь.

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"
#include <Wire.h>
#include <Adafruit_BMP085.h>

#define SENDING_DELAY 1000
#define PHOTO_SENSOR A0
```

```
RF24 radio(9, 10);
Adafruit_BMP085 bmp;
```

Листинг 2.1: Подключение библиотек и инициализация периферии (Устройство 1)

Выбираются канал связи, мощность, скорость передачи данных и идентификационный номер передатчика. Далее в бесконечном цикле loop датчики опрашиваются раз в секунду (SENDING_DELAY), собираются в массив данных типа double и передаются на базовую станцию.

```
void setup(void){
  bmp.begin();

  radio.begin();
  radio.setChannel(0x73);
  radio.setPALevel (RF24_PA_MAX);
  radio.setDataRate (RF24_250KBPS);

  radio.openWritingPipe (0xAABBCCDD11LL);

  radio.powerUp();
  radio.stopListening();
}

void loop(void){
  double Photo_sensor = analogRead(PHOTO_SENSOR);

  double Temp = bmp.readTemperature();
  double Pressure = bmp.readPressure();
  double Altitude = bmp.readAltitude();

  double data[] = {Temp, Pressure, Altitude, Photo_sensor};
  radio.write(&data, sizeof(data));

  delay(SENDING_DELAY);
}
```

Листинг 2.2: Сбор и передача данных (Устройство 1)

2.2.2 Устройство 2

Ко второму устройству подключены датчики DS18B20, DH21 и модуль nRF24l01. Программа немного похожа на предыдущую и выглядит следующим образом: внешние платы инициализируются, собираются данные и посылаются на базовую станцию.

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"

#include <OneWire.h>
#include <DallasTemperature.h>
```

```
#include <DHT.h>

#define SENDING_DELAY 1000
#define TEMP 4
#define DHTPIN 3
#define DHTTYPE DHT21

RF24 radio(9, 10);
byte counter;

OneWire oneWire(TEMP);
DallasTemperature sensor(&oneWire);

DHT dht(DHTPIN, DHTTYPE);

void setup(void){
  sensor.begin();
  dht.begin();

  radio.begin();
  radio.setChannel(0x73);
  radio.setPALevel (RF24_PA_MAX);
  radio.setDataRate (RF24_250KBPS);

  radio.openWritingPipe (0xAABBCCDD22LL);

  radio.powerUp();
  radio.stopListening();
}

void loop(void){
  sensor.requestTemperatures();
  double Temp1 = sensor.getTempCByIndex(0);

  double Temp2 = dht.readTemperature();
  double Humidity = dht.readHumidity();


  double data[] = {Temp1, Temp2, Humidity};
  radio.write(&data, sizeof(data));

  delay(SENDING_DELAY);
}
```

Листинг 2.3: Программа устройства 2

Здесь также можно поменять задержку отправки данных или ножки микроконтроллера, к которому подключены внешние платы.

2.2.3 Базовая станция

Базовая станция состоит из платы Arduino, модуля nRF24l01 и нескольких индицирующих светодиодов разных цветов. Здесь алгоритм работы немного

сложнее, так как работа приемника может вестись как с одним передатчиком, так и с двумя. Разберем код по порядку.

1. Вызовем библиотеки и инициализируем плату передатчика. Объявим переменные data1, data2 типа double для записи полученных данных. Также введем флажки состояний, которые помогут нам в дальнейшем.

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"

#define GREEN 3
#define YELLOW 4
#define RED 5

RF24 radio(9, 10);
uint8_t pipe;

byte count, count1, count2;
bool firstTr = false;
bool secondTr = false;

double data1[] = { 0., 0., 0., 0. };
double data2[] = { 0., 0., 0. };
double trash[] = { 0., 0., 0., 0. };
```

Листинг 2.4: Инициализация базовой станции

2. Настроим приемник на соответствующий канал связи, введем идентификационные номера обоих передатчиков и поприветствуем пользователя миганием светодиодов.

```
void setup() {
  Serial.begin(9600);

  radio.begin();

  radio.setChannel(0x73);
  radio.setPALevel(RF24_PA_MAX);
  radio.setDataRate(RF24_250KBPS);

  radio.openReadingPipe(1, 0xAABBCCDD11LL);
  radio.openReadingPipe(2, 0xAABBCCDD22LL);

  radio.powerUp();
  radio.startListening();

  RYG_Indicator(100);
}
```

Листинг 2.5: Настройка базовой станции

3. Далее в бесконечном цикле loop проверим, есть ли в эфире сигналы передающих станций. Если мы поймали хотя бы один передатчик, то в течении некоторого времени ждем, пока не придет сигнал со второго. Пока ждем, уничтожаем данные, полученные с первого, так как нам нужна информация со второго

передатчика. После получения неоходимого объединим полученные данные и выдадим их в порт. Будет индицировать зеленый светодиод.

```
if (radio.available(&pipe) && !firstTr && !secondTr) {
    if (pipe == 1) {
      radio.read(&data1, sizeof(data1));
      while (pipe != 2) {
        radio.available(&pipe);
        if (pipe == 2) {
          radio.read(&data2, sizeof(data2));
          makeString(data1, data2);

          greenIndicator(50);

          count1 = 0;
          break;
        }
        else {
          radio.read(&trash, sizeof(trash));
        }
        if (count1 > 100) {
          firstTr = true;
          count1 = 0;
          break;
        }
        count1++;
        delay(100);
      }
      pipe = 0;
      count = 0;
    }
```

Листинг 2.6: Получение данных с двух устройств

4. Если после прохождения некоторого времени, мы так и не получили сигнал со второго, то флажки состояний меняются, и мы переходим к другой части кода. Теперь будем работать соответственно с тем передатчиком, который есть в эфире, сигнализировать об этом будет желтый светодиод. Но в то же время мы проверяем, не появился ли второй передатчик в сети, и если да, то флажки состояний меняются и мы переходим к предыдущему коду и работаем с двумя из них. В порт будет выдаваться частично заполненный массив.

```
else if (radio.available(&pipe) && firstTr && !secondTr) {
    if (pipe==1){
      radio.read(&data1, sizeof(data1));
      double data2[] = { 0., 0., 0. };
      makeString(data1, data2);

      yellowIndicator(50);
    }
    else {
      firstTr = false;
      secondTr = false;
    }
    count = 0;
```

```
}
```

Листинг 2.7: Получение данных с одного устройства

5. В ином случае, когда в сети нет ни первого, ни второго передатчика, приемная станция перейдет в режим ожидания и станет индицировать красным светодиодом до тех пор, пока не появится хотя бы один из передатчиков. В этом режиме в порт выдается массив с нулями. После появления хотя бы одного передатчика, флажки состояний поменяются и мы перейдем к одному из верхних уровней кода.

```
while (!radio.available(&pipe) && count > 50) {
    double data1[] = { 0., 0., 0., 0. };
    double data2[] = { 0., 0., 0. };
    makeString(data1, data2);

    redIndicator(500);

    firstTr = false;
    secondTr = false;
}
```

Листинг 2.8: Режим ожидания

Таким образом, в порту всегда будет массив данных: полностью заполненный, если принимаем с двух источников; частично заполненный данными, частично нулями, если с одного приемника и массив нулей, если ничего не принимаем. Окончательный массив будет иметь вид: {BMP180: Temp, Pressure, Alt; Photo_res; DS18B20: Temp; DH21: Temp, Humidity}.

В результате у нас есть приемо-передающая аппаратура. Так, мы можем визуализировать и обрабатывать данные с помощью LabView, чем займемся в следующей главе.

Глава 3

Разработка программы на LabView

3.1 Определение требований к программе

Программа должна иметь возможность установить соединение с метеостанцией, используя соответствующий интерфейс связи, в нашем случае по порту USB, чтобы получать данные о погоде. Она должна предоставлять графический интерфейс пользователя для визуализации и отображения полученных погодных данных. Это могут быть графики, цифровые показатели и другие элементы, которые позволят пользователю наглядно анализировать данные. Программа должна иметь интуитивно понятный и удобный пользовательский интерфейс, который позволит пользователю легко взаимодействовать с метеостанцией, настраивать параметры и анализировать данные. Также предусматривается запись полученных данных в лог файл, с помощью которого можно обрабатывать данные в других программах.

3.2 Пользовательский интерфейс

На пользовательском интерфейсе расположаться:

1. Два индикатора типа Waverform Chart, позволяющие следить за динамикой изменения температуры и влажности;
2. Цифровые индикаторы, показывающие:
   (a) Текущие значения температуры, влажности, давления;
   (b) Высоту местности (посчитанные по давлению), освещенность (в условных единицах);
   (c) Время начала работы программы и время последнего опроса приемника, число проведенных опросов устройства.
3. Слайдер, с помощью которого можно поменять масштабы графиков;

4. Текстовый индикатор, в котором выводятся данные о работающих передатчиках;

5. Поле выбора порта, с которого будут считываться данные (по умолчанию символьная скорость установлена на 9600 бод);

6. Индицирующие светодиоды, дублирующие работу основных, которые расположены на приемнике;

7. Текстовый индикатор, указывающий пусть к лог файлу;

8. Кнопка стоп, останавливающая работу всей программы.

3.3 Считывание и обработка данных

Блок диаграмма программы состоит из нескольких частей:

1. Инициализация программы:
   (a) Запуск считывания порта посредством функции VISA Open;
   (b) Инициализация лог файла;
   (c) Установка начальных параметров графиков.

2. Цикл, в котором происходит основная работа программы:
   (a) Считывание порта посредством функции VISA Read;
   (b) Запись полученных данных в лог файл;
   (c) Дальнейшая обработка полученных данных - разбитие массива на состовляющие и вывод значений на график/цифровой индикатор при каждом поступлении неповторяющихся данных;
   (d) Обработка статуса приемника и изменения параметра слайдера.

3. Завершающая работу часть программы, происходит при нажатии кнопки стоп:
   (a) Сохранение и закрытие лог файла;
   (b) Остановка считывания порта посредством VISA Close.

Глава 4

Тестирование и результаты

4.1 Проверка функциональности

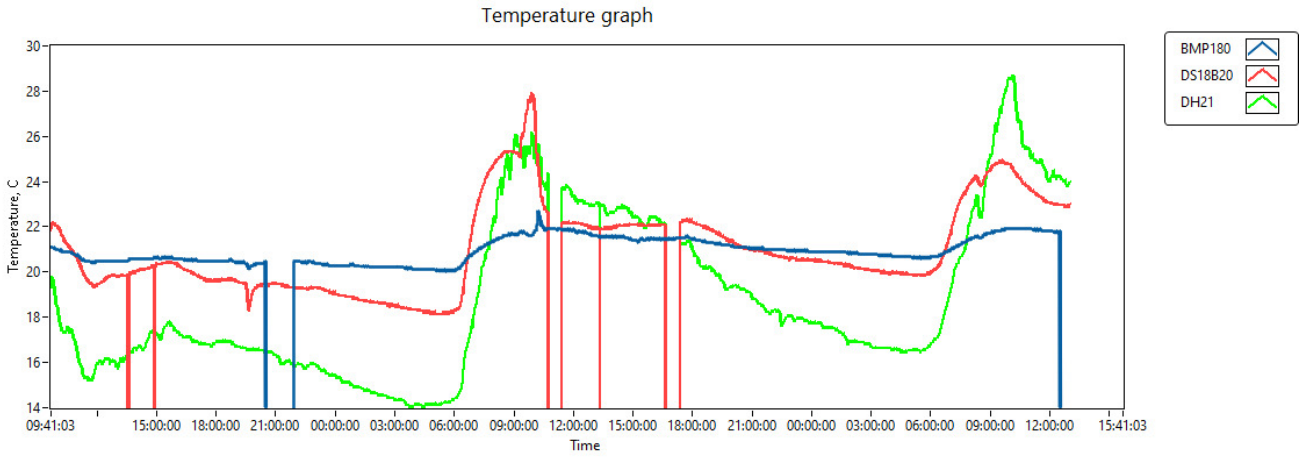

Рис. 4.1: Изменение температуры за два дня

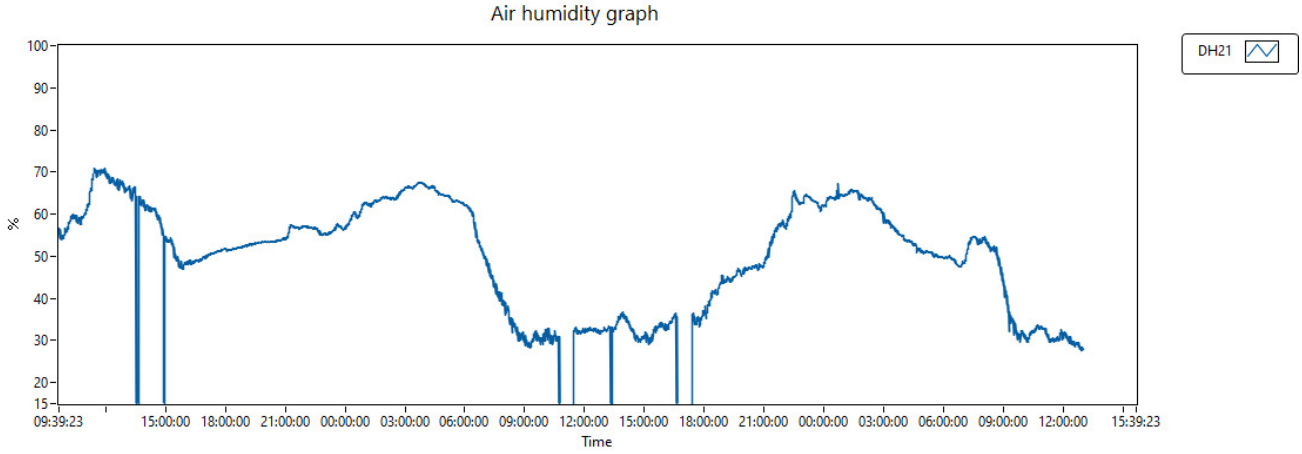

Рис. 4.2: Изменение влажности за два дня

После разработки метеостанции и программы на LabView была проведена проверка функциональности. Метеостанция успешно подключается к компьютеру и программе, все необходимые данные корректно считываются и отображаются на соответствующих полях в пользовательском интерфейсе.

Программа позволяет выбирать временные промежутки и следить за динамикой изменения как температуры, так и влажности, что довольно таки удобно для анализирования. Она позволяет сохранить как график, так и данные в необходимом пользователю месте. Стабильность программы тоже успешно проверена.

4.2 Анализ полученных результатов

В ходе тестирования, метеостанция и программа работали практически непрерывно, как можно заметить на рис. 4.1 и рис. 4.2. Проседание значений до нуля связана с тестовым отключением одного из передатчиков. Как видно, приемник и программа способна работать и с одним из них. После включения предатчика, их парная работа продолжается. Индикаторы показывают текущую температуру, влажность, давление, высоту, освещенность (рис. 4.3) и состояние приборов.

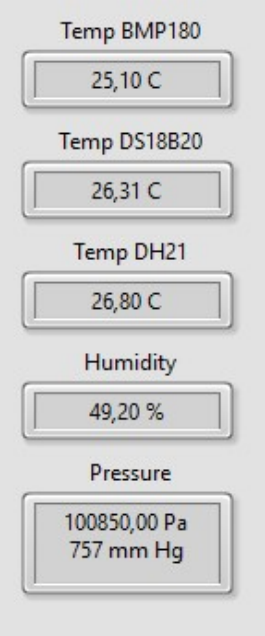

Рис. 4.3: Индикаторы температуры, влажности и давле-

Title

```
1          |          BMP180          |       | DS18 |   DH21    |
2          | Temp   Pressure    Alt |  Ph_r | Temp |Temp  Humid|
3 [9:39:33] 21.10, 100558.00, 63.64, 168.00, 21.75, 19.10, 56.00
4 [9:39:36] 21.10, 100554.00, 64.47, 168.00, 21.75, 19.10, 55.90
5 [9:39:38] 21.10, 100555.00, 64.56, 169.00, 21.69, 19.10, 56.00
6 [9:39:40] 21.10, 100557.00, 63.72, 169.00, 21.75, 19.10, 55.90
```

Рис. 4.4: Лог файл

На рис. 4.4 показан пример лог файла. Он является важной составляющей системы метеостанции, который используется для записи и хранения данных о погодных параметрах в каждый момент времени. Структура лог файла:

1. Заголовок файла, который описывает содержание файла и датчик, с которого были получены данные;
2. Запись данных, которая представляет собой момент времени и соответствующие значения погодных параметров. В нашем случае, запись содержит следующую информацию:

(a) Дата и время: Временная метка, указывающая момент, когда были сняты данные.

(b) Температура (Temp): Значение температуры в указанный момент времени.

(c) Влажность (Humid): Значение влажности в указанный момент времени.

(d) Давление (Pressure): Значение давления в указанный момент времени.

(e) Высота (Alt): Значение высоты над уровнем моря в указанный момент времени.

(f) Освещенность (Ph_r): Значение освещенности или интенсивности света в указанный момент времени.

Заключение и итоги работы

В рамках данной работы была разработана метеостанция на базе Arduino с реализацией mesh-сети и программы на LabView для управления и анализа погодных данных. В процессе выполнения работы были достигнуты следующие результаты:

1. Разработана аппаратная часть метеостанции, включающая Arduino и соответствующие датчики для сбора данных о погоде. Были настроены необходимые компоненты, обеспечивающие надежный сбор и передачу данных.

2. Реализована mesh-сеть с использованием передатчиков, что позволяет обеспечить беспроводную передачу данных между узлами метеостанции. Это обеспечивает расширяемость и гибкость системы.

3. Разработана программа на LabView, которая обеспечивает управление метеостанцией, визуализацию погодных данных и анализ полученной информации. Пользовательский интерфейс программы позволяет легко настраивать параметры метеостанции и анализировать данные в удобной форме.

4. Проведена проверка функциональности разработанной метеостанции и программы на LabView. Проверка включала подключение к метеостанции, получение данных, визуализацию и анализ данных. Результаты проверки подтвердили правильную работу системы и соответствие требованиям.

В ходе выполнения работы были выявлены некоторые возможности для улучшения и расширения системы. Например, можно добавить дополнительные датчики для сбора дополнительных погодных параметров или расширить функциональность программы на LabView для более сложного анализа данных. Можно также увеличить количество узлов mesh-сети.

В целом, данная работа является успешным шагом в разработке метеостанций с использованием Arduino и программного обеспечения на LabView. Полученные результаты могут быть полезны для дальнейших исследований в области метеорологии, а также для создания практических приложений в сфере агрокультуры, экологии, строительства и других областей, где наблюдение и анализ погодных условий имеют важное значение.

Литература

[1] И.В.Федосов. *Основы программирования в LabVIEW*

[2] Стивен Прата. *Язык программирования C++. Лекции и упражнения*

[3] Сайт Амперка [Электронный ресурс] `https://amperka.ru/`

[4] Сайт Амперкот [Электронный ресурс] `https://amperkot.ru/`

Приложения

Схема подключения датчиков и передатчиков

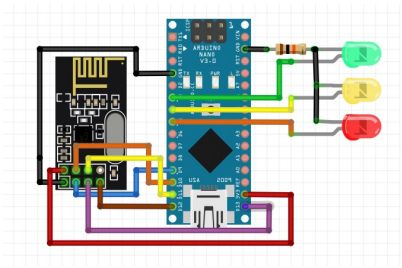

Рис. 4.5: Схема подключения базовой станции

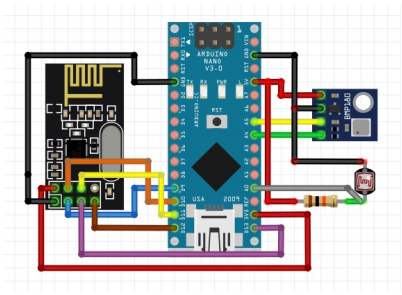

Рис. 4.6: Схема подключения первой станции

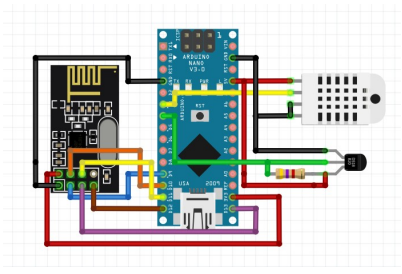

Рис. 4.7: Схема подключения второй станции