Терморезистор NTC 10 К MF52

Терморезистор (или термистор) — это такой резистор, который меняет свое электрическое сопротивление в зависимости от температуры.

Существует два вида термисторов: PTC — с положительным температурным коэффициентом, и NTC — с отрицательным. Положительный коэффициент означает, что с повышением температуры сопротивление термистора растёт. NTC-термистор ведет себя противоположным способом.

Также термисторы отличаются номинальным сопротивлением, которое соответствует комнатной температуре — 25 С°. Например, популярными являются термисторы с номиналом 100 кОм и 10 кОм. Такие термисторы часто используют в 3D-принтерах.

В этом уроке мы будет использовать термистор NTC 100K. Вот такой:

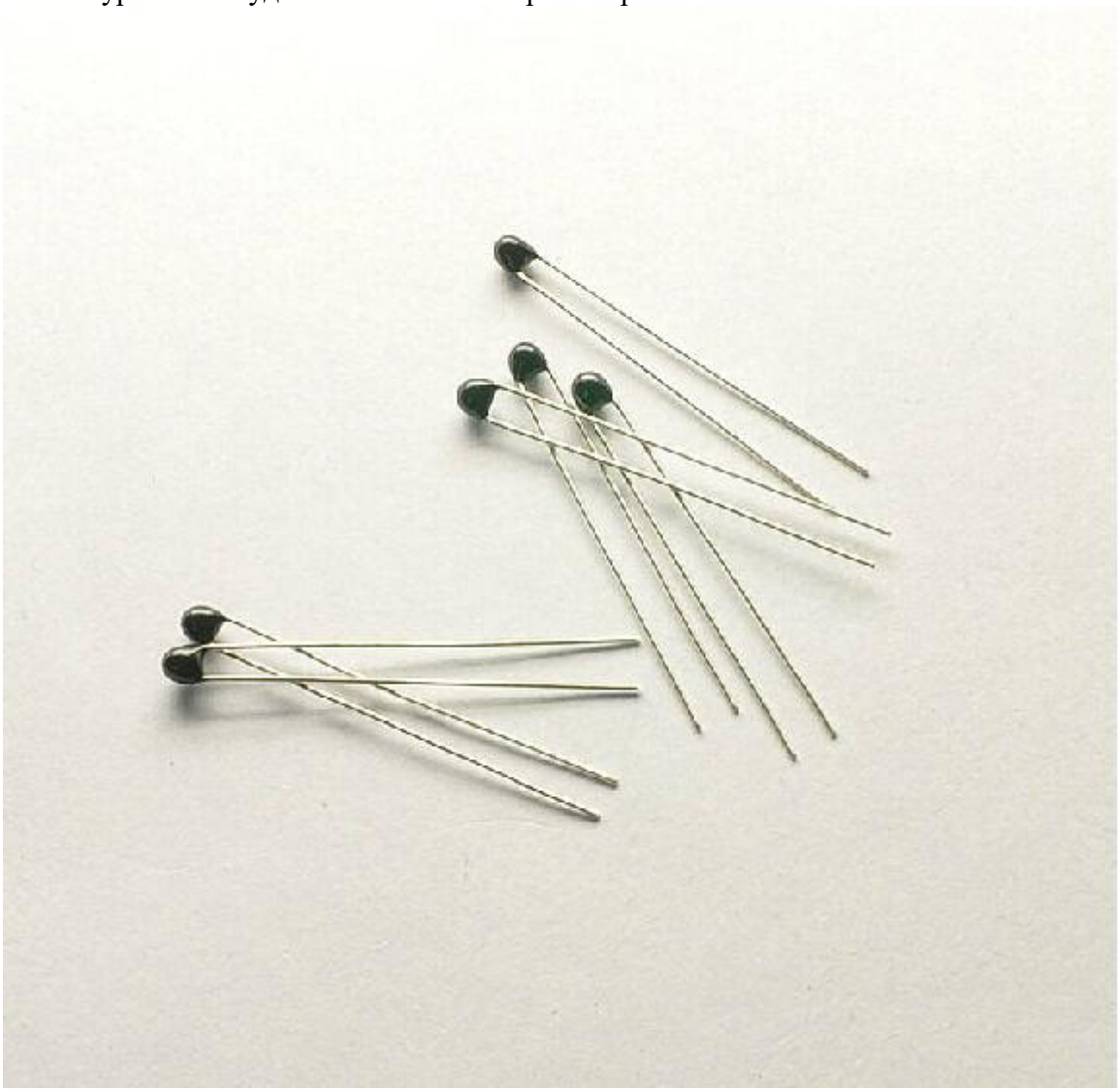

Рисунок 1. NTC 100K

1. Подключение термистора к Ардуино

Чтобы измерить сопротивление термистора, подключим его в качестве нижнего плеча делителя напряжения. Среднюю же точку делителя подключим к аналоговому входу Ардуино — A0. Подобный способ использовался в уроке про фоторезистор.

Принципиальная схема

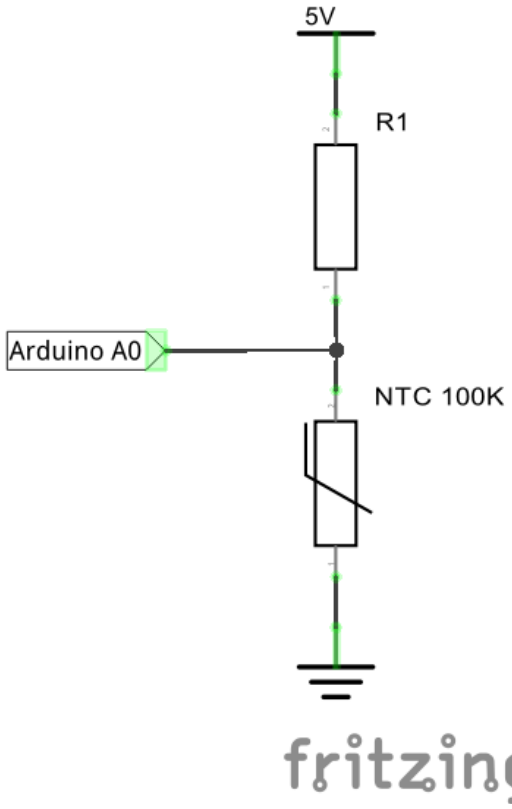

Рисунок 2. Принципиальная схема подключения NTC

Схема подключения термистора NTC 100K к Ардуино

Внешний вид макета

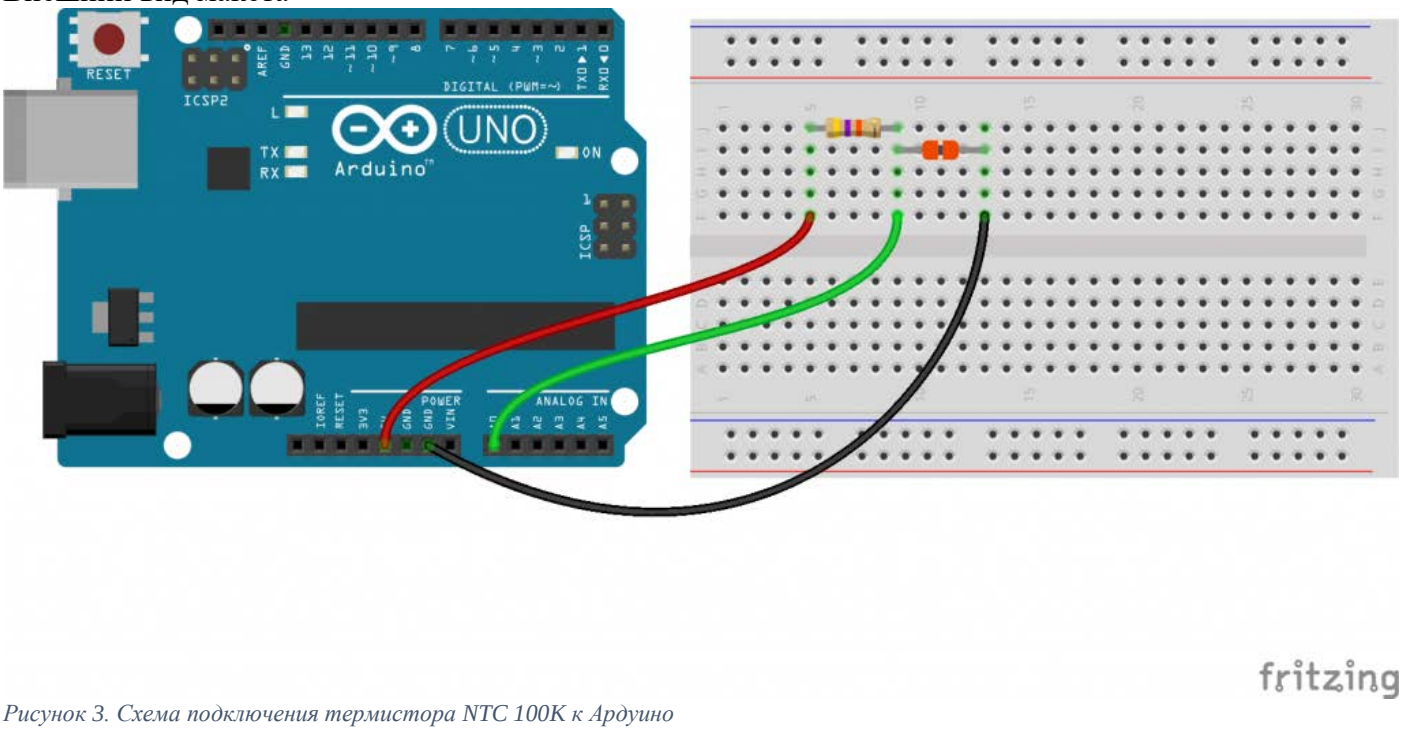

Рисунок 3. Схема подключения термистора NTC 100K к Ардуино

Какое сопротивление должен иметь резистор в верхнем плече делителя? Как правило, используют резистор с сопротивлением, совпадающим по порядку с номиналом термистора. В нашем уроке мы используем резистор на R1 = 102 кОм, его легко получить последовательным соединением двух резисторов на 51 кОм.

2. Программа для вычисления сопротивления термистора

Первая программа, которую мы напишем, будет вычислять сопротивление термистора в Омах.

Листинг 1. вычислять сопротивление термистора в Омах.

```
#define SERIAL_R 102000 // сопротивление последовательного резистора, 102 кОм
const byte tempPin = A0;
void setup() {
    serial.begin( 9600 );
    pinMode( tempPin, INPUT );
}
void loop() {
    int t = analogRead( tempPin );
    float tr = 1023.0 / t - 1;
    tr = SERIAL_R / tr;
    serial.println(tr);
    delay(100);
}
```

Результат работы программы:

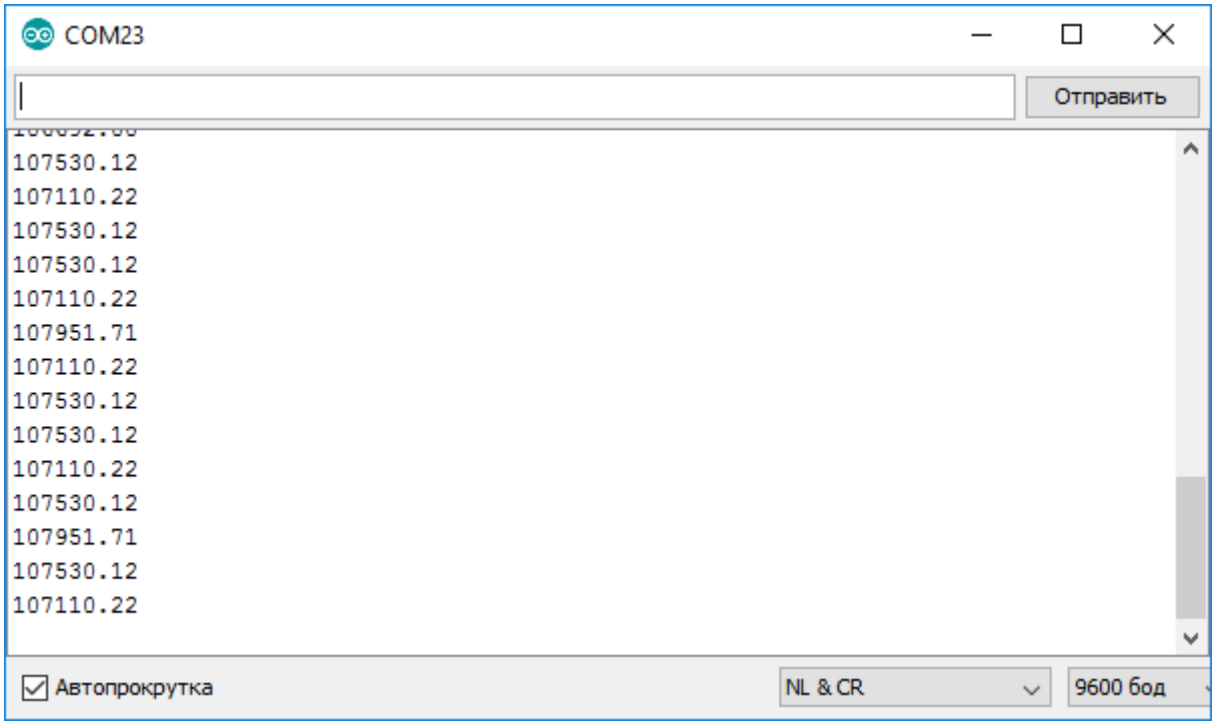

Рисунок 4. Схема подключения термистора NTC 100K к Ардуино

Можно заметить, что измеренное сопротивление термистора меньше 100 кОм, значит температура окружающей среды ниже 25 С°. Следующий шаг — вычисление температуры в градусах Цельсия.

3. Программа для вычисления температуры на термисторе

Чтобы вычислить значение температуры используют формулу Стейнхарта — Харта:

$$\frac{1}{T} = A + B\ln(R) + C(\ln(R))^3$$

Рисунок 5. Уравнение Стейнхарта — Харта

Уравнение имеет параметры A,B и C, которые нужно брать из спецификации к датчику. Так как нам не требуется большой точности, можно воспользоваться модифицированным уравнением (B-уравнение):

$$\frac{1}{T} = \frac{1}{T_0} + \frac{1}{B}\ln\left(\frac{R}{R_0}\right)$$

Рисунок 6. Модифицированное уравнение Стейнхарта — Харта

В этом уравнении неизвестным остается только параметр B, который для NTC термистора равен 3950. Остальные параметры нам уже известны:

T0 — комнатная температура в Кельвинах, для которой указывается номинал термистора; T0 = 25 + 273.15;
T — искомая температура, в Кельвинах;
R — измеренное сопротивление термистора в Омах;
R0 — номинальное сопротивление термистора в Омах.

Модифицируем программу для Ардуино, добавив расчет температуры:

Листинг 2. Программа для рассчёта температуры

```
#define B 3950 // B-коэффициент
#define SERIAL_R 102000 // сопротивление последовательного резистора, 102 кОм
#define THERMISTOR_R 100000 // номинальное сопротивления термистора, 100 кОм
#define NOMINAL_T 25 // номинальная температура (при которой TR = 100 кОм)

const byte tempPin = A0;

void setup() {
    Serial.begin( 9600 );
    pinMode( tempPin, INPUT );
}

void loop() {
    int t = analogRead( tempPin );
    float tr = 1023.0 / t - 1;
    tr = SERIAL_R / tr;
    Serial.print("R=");
    Serial.print(tr);
    Serial.print(", t=");

    float steinhart;
    steinhart = tr / THERMISTOR_R; // (R/Ro)
    steinhart = log(steinhart); // ln(R/Ro)
    steinhart /= B; // 1/B * ln(R/Ro)
    steinhart += 1.0 / (NOMINAL_T + 273.15); // + (1/To)
    steinhart = 1.0 / steinhart; // Invert
    steinhart -= 273.15;
    Serial.println(steinhart);

    delay(100);
}
```

Результат:

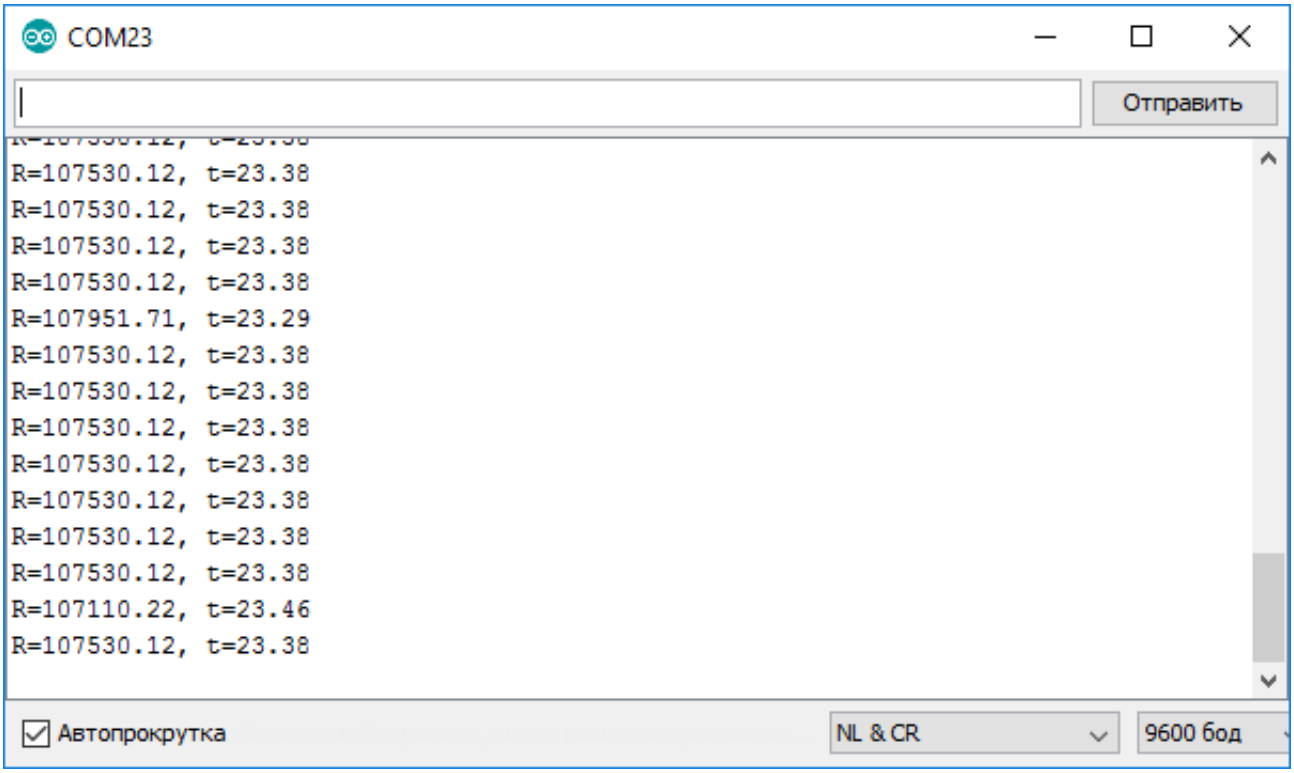

Рисунок 7. Значения температуры на термисторе, Ардуино

Значения температуры на термисторе, Ардуино

Уже лучше! Программа показывает нам температуру в градусах Цельсия. Как и ожидалось, она немного ниже 25 С°.

Теперь можно изменить проект.
Оставьте схему подключения прежнюю, но загрузите новый код:

Листинг 3. Программа чтения данных с термистора

```
// Программа чтения данных с термистора
const int POT=0; // Аналоговый вход 0 для подключения потенциометра
int val =0; // Переменная для хранения значения потенциометра
void setup()
{
Serial.begin(9600);
}
void loop()
{
val = analogRead(POT);
Serial.println(val);
delay(500);
}
```

Подробно функционирование последовательного интерфейса обмена данными мы рассмотрим в последующих главах. А сейчас достаточно знать, что сначала необходимо инициировать последовательное соединение, вызвав функцию Serial.begin(), единственный аргумент которой задает скорость передачи данных в бодах. Скорость передачи данных определяет количество битов, передаваемых в секунду. Высокая скорость передачи позволяет передавать больше данных за меньшее время, но может привести к ошибкам в некоторых системах связи. В наших примеэах выбрана скорость 9600 бод.

В каждой итерации цикла переменная val получает аналоговое значение, считанное командой analogRead() с входа, соединенного с входом A0. Далее это значение функция Serial.println() выводит в последовательный порт, соединенный с компьютером. Затем следует задержка в полсекунды (чтобы числа выводились не быстрее, чем вы можете их прочитать).
После загрузки на плату Arduino вы заметите, что светодиод TX, расположенный на плате, мигает каждые 500 мс (по крайней мере, так должно быть). Этот индикатор показывает, что плата Arduino передает данные через последовательный USB-интерфейс на компьютер. Для просмотра данных подойдут любые терминальные программы, но в Arduino IDE есть встроенный монитор последовательного порта, для запуска которого нажмите кнопку, обведенную красным кружком на рис. 8.

Рисунок 8. Кнопка запуска монитора последовательного порта

После запуска монитора последовательного порта на экране компьютера появляется окно с отображением потока передаваемых чисел. Нагрейте термистор пальцами, и вы увидите, что выводимые значения меняются. Если нагревать, числа начинают уменьшаться, если охлаждать — увеличиваются. Пример отображения данных показан на рис. 9.

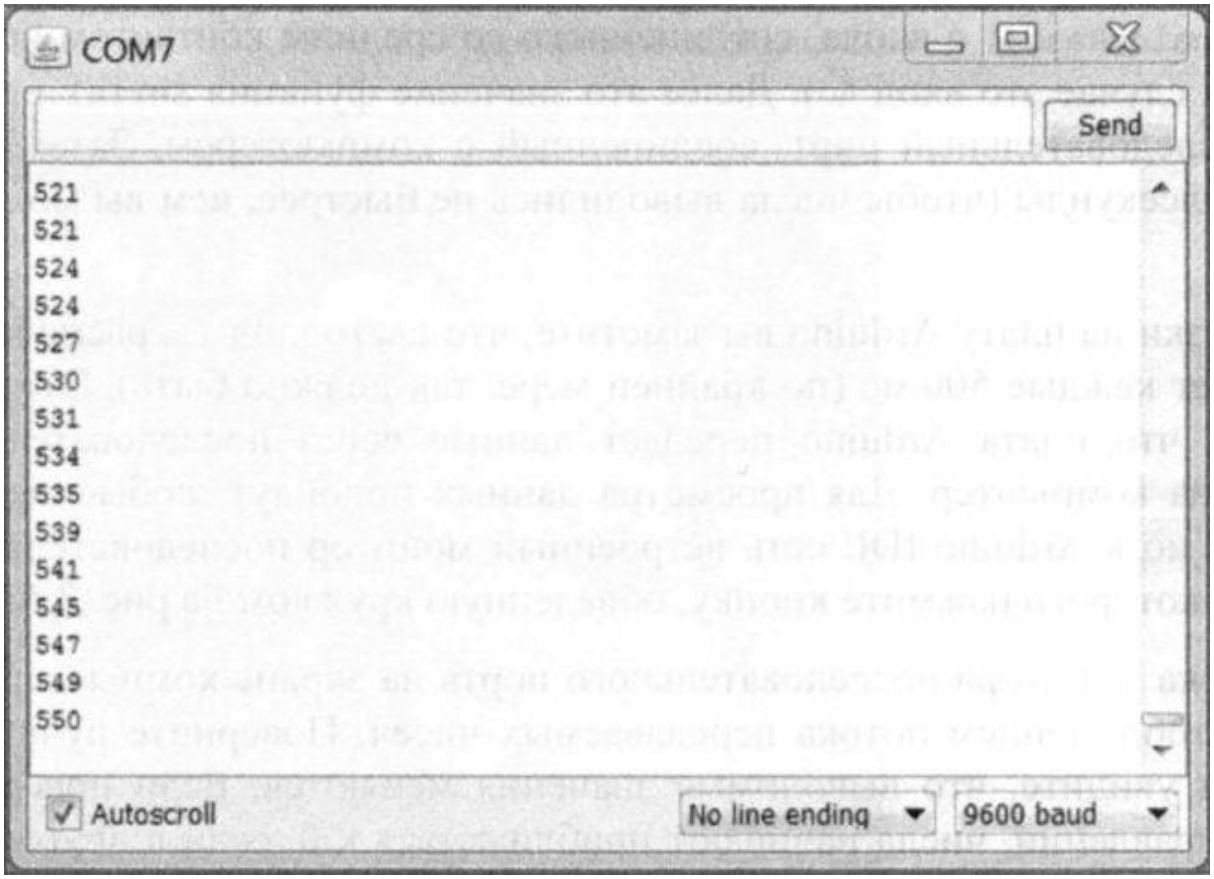

Рисунок 9. Вывод данных в последовательный порт

ПРИМЕЧАНИЕ

Если выводятся непонятные символы, убедитесь, что скорость передачи данных установлена правильно. В программе порт инициализирован на скорость 9600 бод, такое же значение необходимо установить в настройках монитора последовательно порта.

Работа с аналоговым датчиком температуры

Рассмотрим простой пример работы с датчиком температуры, упомянутым в предыдущем разделе. Вы можете выбрать любой аналоговый датчик из приведенного ранее списка или взять какой-нибудь другой. Последовательность действий, описанная далее, практически одинакова для любого аналогового датчика.

Для начала подсоедините к плате Arduino Uno RGB-светодиод, а датчик температуры оставьте без изменений (Рисунок 10).

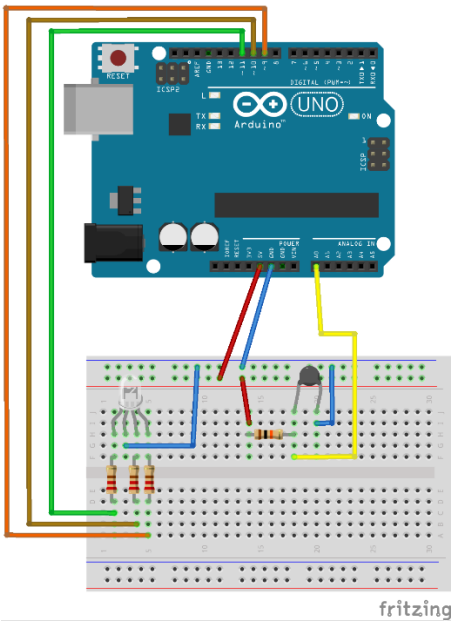

Рисунок 10. Схема подключения датчика температуры

На основе этой схемы создадим простую систему, сигнализирующую об изменении температуры. RGB-светодиод будет гореть зеленым, когда температура находится в пределах допустимого диапазона, красным, когда станет жарко, и синим, когда становится холодно.
Прежде всего, определите приемлемый для вас температурный диапазон. Используя программу из листинга 3, определите аналоговые значения для верхнего и нижнего порогов температуры. Для меня нижний порог комфортной температуры составляет 20 °C, что соответствует аналоговому значению 143. У вас эта цифра может быть другой. Следите за показаниями в мониторе последовательного порта при наступлении нижнего и верхнего предела температуры. Эти значения можно из формулы связывающей температуру (в °C) с входным напряжением (в мВ):

Температура (°C) × 10 = Напряжение (мВ) — 500.

Напряжение 700 мВ соответствует температуре 20°C. Расчет по формуле (или просто анализ показаний монитора последовательного порта) дает для 22°C цифровое значение 139, для 18°C — 147. Эти величины выберем для нижнего и верхнего значений комфортной температуры, чтобы изменять цвет светодиода. Функция analogRead() будет считывать показания датчика температуры, digitalWrite() — устанавливать цвет светодиода.

СОВЕТ

Рекомендую вам не копировать листинг 4, а попробовать написать текст программы самостоятельно, чтобы убедиться в своих силах. Сравните свой результат с приведенным далее.

Листинг 4. Программа температурного оповещателя — tempalert.ino

```
// Температурный оповещатель
const int  BLED=9;                //Контакт 9 для вывода BLUE RGB-светодиода
const int  GLED=10;               //Контакт 9 для вывода GREEN RGB-светодиода
const int  RLED=11;               //Контакт 9 для вывода RED RGB-светодиода
const int  TEMP=0;                //Контакт A0 для подключения датчика температуры
const int LOWER_BOUND=147;        //Нижний порог
const int UPPER_BOUND=139;        //Верхний порог
int val =0;                       //Переменная для чтения аналогового значения
void setup()
{
pinMode (BLED, OUTPUT);           //Сконфигурировать BLUE контакт светодиода
                                  //как выход
pinMode (GLED, OUTPUT);           //Сконфигурировать GREEN контакт светодиода
//как выход
pinMode (RLED, OUTPUT);           //Сконфигурировать RED контакт светодиода
                                  //как выход
}
void loop()
{
val = analogRead(TEMP);
if (val > LOWER_BOUND)
{
digitalWrite(RLED, LOW);
digitalWrite(GLED, LOW);
digitalWrite(BLED, HIGH);
}
else if (val < UPPER_BOUND)
{
digitalWrite(RLED, HIGH);
digitalWrite(GLED, LOW);
```

```
digitalWrite(BLED, LOW);
}
else
{
digitalWrite(RLED, LOW);
digitalWrite(GLED, HIGH);
digitalWrite(BLED, LOW);
}
}
```