Практическая работа 17

Комнатный термометр

В этом эксперименте мы измеряем температуру окружающей устройство среды и с помощью шкалы показываем, на сколько она превышает заданный порог.

Термистор — это переменный резистор, который меняет собственное сопротивление в зависимости от температуры. Это термистор B57164-K 103-J, сопротивление которого при 25 °C равно 10 кОм. По мере увеличения температуры сопротивление падает, по мере уменьшения — растёт.

На его основе очень просто создать схему, которая бы поставляла данные о температуре окружающей среды в виде аналогового сигнала на управляющую электронику. Для этого достаточно сделать элементарный делитель напряжения, где одним из резисторов будет этот, а вторым — резистор на 10 кОм.

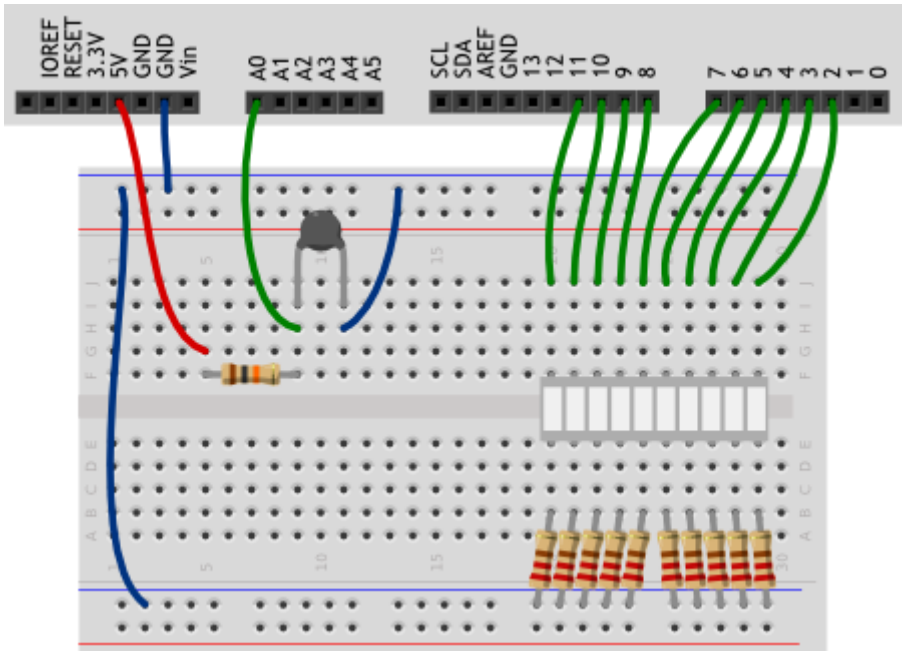
Задание 1. Ответить на вопросы

1. Что такое термистр?
2. К какому порту: аналоговому или цифровому – необходимо подключить термистр?

Задание 2. Список деталей для эксперимента

1. 1 плата Arduino Uno
2. 1 беспаечная макетная плата
3. 1 светодиодная шкала
4. 10 резисторов 220 Ом
5. 14 проводов «папа-папа»
6. 1 термистор
7. 1 резистор 10 Ком

Схема на макетной плате:



1. Зарисуйте принципиальную схему установки.

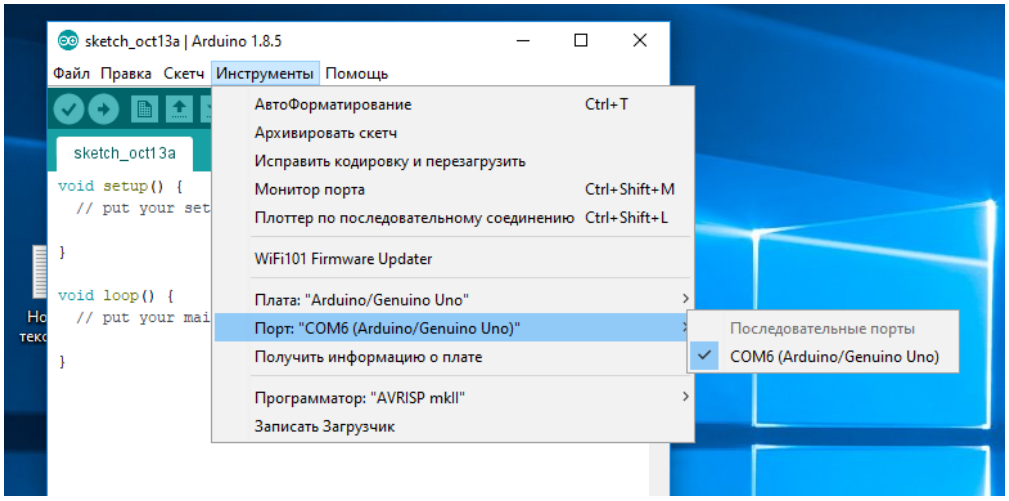
Обратите внимание

Термистор мы включили в известную нам схему делителя напряжения.

Задание 3. Программирование микроконтроллера

1. Запустите приложение

2. Убедитесь, что выбран нужный порт



3. Наберите в редакторе кода следующий код программы:

```
// Огромное количество готового кода уже написано другими людьми
// и хранится в виде отдельных файлов, которые называются
// библиотеками. Для использования кода из библиотеки, её нужно
// подключить (англ. include). Библиотека «math» даёт разные
// математические функции, в том числе функцию логарифма
// (англ. log), которая нам понадобится далее
#include <math.h>

#define FIRST_LED_PIN 2
#define LED_COUNT     10

// Параметр конкретного типа термистора (из datasheet):
#define TERMIST_B 4300

#define VIN 5.0

void setup()
{
  for (int i = 0; i < LED_COUNT; ++i)
    pinMode(i + FIRST_LED_PIN, OUTPUT);
}

void loop()
{
  // вычисляем температуру в °C с помощью магической формулы.
  // Используем при этом не целые числа, а вещественные. Их ещё
  // называют числами с плавающей (англ. float) точкой. В
  // выражениях с вещественными числами обязательно нужно явно
  // указывать дробную часть у всех констант. Иначе дробная
  // часть результата будет отброшена

   float voltage = analogRead(A0) * VIN / 1023.0;
   float rl = voltage / (VIN - voltage);


   float temperature = 1./( 1./(TERMIST_B)*log(rl)+1./(25. + 273.)

  for (int i = 0; i < LED_COUNT; ++i) {
    // при 21°C должен гореть один сегмент, при 22°C — два и
    // т.д. Определяем должен ли гореть i-й нехитрым способом
    boolean enableSegment = (temperature >= 21+i);
    digitalWrite(i + FIRST_LED_PIN, enableSegment);
  }
}
```

Пояснения к коду

Директивы для подключения библиотек #include включаются в начало программы.

В этом эксперименте мы подключаем библиотеку math.h для того, чтобы использовать функцию взятия натурального логарифма x log(x).

В переменных типа float можно хранить дробные числа, числа с плавающей точкой.

При использовании переменных данного типа имейте в виду:

✓ при операциях с их использованием, указывайте нулевую дробную часть у целых констант, как в примере

✓ они могут принимать значения от -3.4028235×1038 до 3.4028235×1038,

✓ при этом количество значащих цифр может быть 6-7: всех цифр, не только после запятой!

✓ точность вычислений с такими данными невелика, у вас могут возникнуть неожиданные ошибки, например, при использовании float в условном операторе. Не полагайтесь на точность!

✓ вычисления с float происходят медленнее, чем с целыми числами

Показания термистора связаны с температурой нелинейно, поэтому нам приходится использовать такую громоздкую формулу.

Задание 4. Ответьте на следующие вопросы

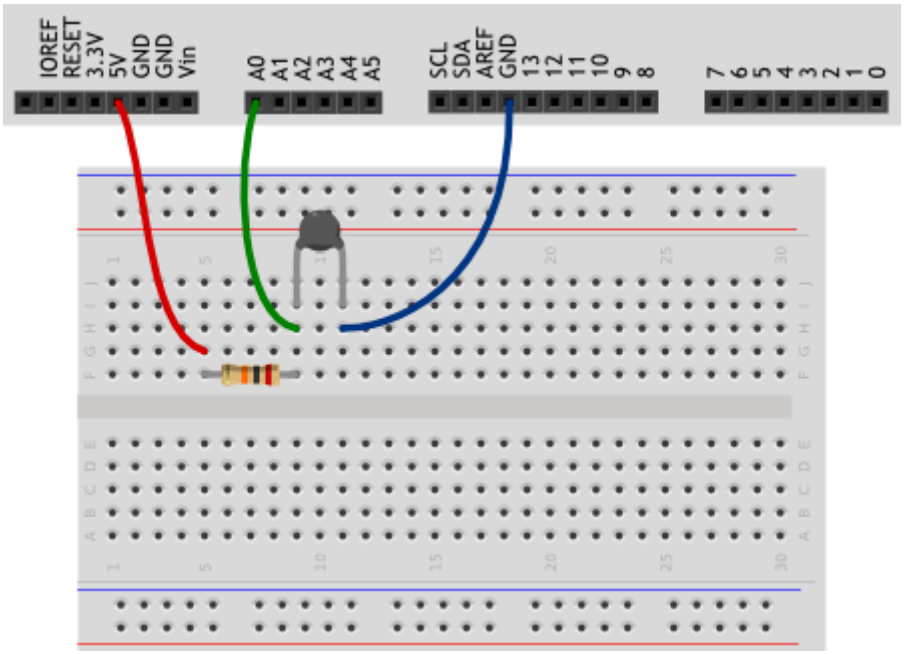
1. Как нужно подключить термистор, чтобы получать на Arduino данные о температуре?
2. Каким образом можно воспользоваться ранее разработанными функциями, не переписывая их в программный код?
3. Чем неудобно использование чисел с плавающей точкой на Arduino?
4. Что за выражение стоит справа от `=` при объявлении булевой переменной `enableSegment`?

Задание 5. Самостоятельно измените существующую программу и схему

1. Измените код программы таким образом, чтобы индикатор включался при 0 градусов и его показания прирастали на одно деление каждые 5 градусов.
2. Добавьте в схему пьезопищалку и доработайте программу так, чтобы срабатывала звуковая сигнализация при достижении температуры, например, 20 градусов.

В этом эксперименте мы передаем данные об измерениях температуры на компьютер (например, для последующей обработки).

Задание 6. Соберите следующую схему



Зарисуйте принципиальную схему установки.

1. Наберите в редакторе кода следующий код программы:

Пояснения к коду

Очень часто бывает полезно обмениваться данными, например, с компьютером. В частности, для отладки работы устройства: можно, например, смотреть, какие значения принимают переменные.

В данном эксперименте мы знакомимся со стандартным объектом Serial, который предназначен для работы с последовательным портом (UART) Arduino, и его методами (функциями, созданными для работы с данным объектом) begin(), print() и println(), которые вызываются после точки, идущей за именем объекта:

чтобы обмениваться данными, нужно начать соединение, поэтому Serial.begin(baudrate)вызывается в setup()

Serial.print(data) отправляет содержимое data. Если мы хотим отправить текст, можно просто заключить его в пару двойных кавычек: "". Кириллица, скорее всего, будет отображаться некорректно.

Serial.println(data) делает то же самое, только добавляет в конце невидимый символ новой строки.

В print() и println() можно использовать второй необязательный параметр: выбор системы счисления, в которой выводить число (это может быть DEC, BIN, HEX, OCT для десятичной, двоичной, шестнадцатеричной и восьмеричной систем счисления соответственно) или количество знаков после запятой для дробных чисел.

Например,

```
Serial.println(18,BIN);
Serial.print(3.14159,3);
```

в мониторе порта даст результат

```
10010
3.142
```

Монитор порта, входящий в Arduino IDE, открывается через меню Сервис или сочетанием клавиш Ctrl+Shift+M. Следите за тем, чтобы в мониторе и в скетче была указана одинаковая скорость обмена данными, baudrate. Скорости 9600 бит в секунду обычно достаточно. Другие стандартные значения можете посмотреть в выпадающем меню справа внизу окна монитора порта.

Вам не удастся использовать цифровые порты 0 и 1 одновременно с передачей данных по последовательному порту, потому что по ним также идет передача данных, как и через USB-порт платы.

При запуске монитора порта скетч в микроконтроллере перезагружается и начинает работать с начала. Это удобно, если вам нельзя упустить какие-то данные, которые начинаю передаваться сразу же. Но в других ситуациях это может мешать, помните об этом нюансе!

Если вы хотите читать какие-то данные в реальном времени, не забывайте делать delay() хотя бы на 100 миллисекунд, иначе бегущие числа в мониторе будет невозможно разобрать. Вы можете отправлять данные и без задержки, а затем, к примеру, скопировать их для обработки в стороннем приложении.
Последовательность \t выводится как символ табуляции (8 пробелов с выравниванием). Также вы можете использовать, например, последовательность \n для перевода строки. Если вы хотите использовать обратный слеш, его нужно экранировать вторым таким же: \\.

Задание 7. Ответьте на следующие вопросы

1. Какие действия нужно предпринять, чтобы читать на компьютере данные с Arduino?
2. О каких ограничениях не следует забывать при работе с последовательным портом?
3. Как избежать ошибки в передаче данных, содержащих обратный слэш (\)?

Задание 8. Самостоятельно измените существующую программу и схему

1. Перед таблицей данных о температуре добавьте заголовок (например, "Meteostation").
2. Добавьте столбец, содержащий количество секунд, прошедших с момента запуска микроконтроллера. Можно уменьшить интервал передачи данных.