УДК 004

Управление скоростью вращения вентилятора через широтно-импульсную модуляцию на платформе Arduino

Болтовский Гавриил Александрович
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
В данной статье описывается процесс создание устройства, предназначенного для управления скоростью вращения вентиляторов. Результатом исследования станет готовое устройство, с подробным описанием его реализации.
Ключевые слова: PWM, Arduino, управление вентилятором.

Fan speed control with PWM on Arduino platform

Boltovsky Gavriil Alexandrovich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes the process of creating a fan speed control device. The result of the research will be a finished device with a detailed description of its implementation.
Keywords: PWM, Arduino, fan control.

1. Введение

1.1 Актуальность исследования

На Arduino платах все сигнальные входы можно разделить на три группы: аналоговые, цифровые, цифровые с генерацией ШИМ сигнала. ШИМ (широтно-импульсная модуляция) предполагает процесс управления мощностью, посредством включения-выключения потребителя энергии. Область применения огромна, таким образом, можно управлять яркостью светодиодов, мощностью обогревателя, скоростью вращения вентилятора.

Управление скоростью вращения вентилятора через ШИМ наиболее удачно, так как процесс включения-выключения незаметен из-за инертности конструкции вентилятора. Со светодиодами на определённой частоте человеку становится заметно мерцание, что может негативно повлиять на зрение человека при постоянном воздействии.

В подавляющем большинстве случаев вентиляторы работают от 12 вольт постоянного тока, в статье рассмотрен пример работы платы от такого напряжения.

1.2 Обзор исследований

В статье А.А. Мишенина, И.А. Лихолетова [1] описывается преимущества использования широтно-импульсной модуляции в современных бытовых приборах, приводятся основные характеристики ШИМ. Описываются два способа его получения, такие как аналоговый и цифровой. В.П. Беляев в статье [2] рассматривает способ оптимизации работы асинхронного электродвигателя посредством управления через ШИМ, что делает двигатель более мощным, упрощает управление, уменьшает нагрев двигателя, расширяет скоростной диапазон. В статье [3] К.В. Богданова, А.Н. Ловчикова изучаются проблемы EDA систем, где нет отлаженной механизма для создания математических моделей проектируемых устройств. Рассматриваются различные подходы к проектированию EDA, создание новой системы на языке Erlang, в которой возможно создание модульной системы, благодаря ей становится возможным моделирование устройств с преобразователями напряжения с ШИМ. Имитационное моделирование ШИМ регулятора описано в статье [4] В.П. Бабенко и В.К. Битюкова. А.Ф. Газизулин [5] описывает процесс управления освещением посредством ШИМ, исходя из значений на фоторезисторе.

1.3 Цель исследования

Цель исследования – создать устройство для управления скоростью вращения вентилятора с PWM контактом, работающего при напряжении от 5 до 12 вольт с возможность вывода показателей на дисплей.

1.4 Постановка задачи

Для этого необходимо создать схему подключения и написать прошивку, реализующую такой функционал.

2. Методы исследования

Устройство будет собрано на базе Arduino UNO, но возможно использование Arduino Nano, что делает устройство более компактным. Показания будут выводиться на жидкокристаллический дисплей на 2 строки по 16 символов, в качестве элемента управления будет выступать потенциометр. Для питания вентилятора на 12 вольт, потребуется отдельный источник тока, так как USB порт компьютера не способен выдавать более 5 вольт. При подключении дисплея необходимо использовать резистор на 2 Ком.

Arduino платы дают огромную свободу при подключении различных модулей. Можно выбирать к каким входам будет подключен конкретно данный модуль, поэтому однозначной макетной схемы приведено не будет, ниже приведена таблица для подключения модулей в более общем виде (табл. 1).

Таблица 1 – Подключение модулей

Название модуля	**Контакты модуля**	**Соответствующие контакты на Arduino**
Жидкокристаллический дисплей 1602	1. VSS (ground)	GND
	2. VDD (+Ve)	+5V
	3. VE (Contrast v.)	GND через резистор на 2 Ком
	4. RS (Register select)	0_>RX
	5. RW (read/write)	GND
	6. Enable	1_>TX
	7. Data 0	Не используются
	8. Data 1	
	9. Data 2	
	10. Data 3	
	11. Data 4	Цифровой вход
	12. Data 5	Цифровой вход
	13. Data 6	Цифровой вход
	14. Data 7	Цифровой вход
	15. Blinking Anode (+Ve)	+5V
	16. Blinking Cathod (GND)	GND
Потенциометр	Крайний	+5V
	Центральный	Аналоговый вход
	Крайний	GND
Вентилятор	1. Чёрный	-12V
	2. Желтый	+12V
	3.Зелёный (счёт оборотов)	Цифровой вход с подтяжкой через 10 Ком резистор к 5 вольтам
	4. Синий (PWM control)	Цифровой с ШИМ

Собранное устройство изображено на рисунке (рис. 1).

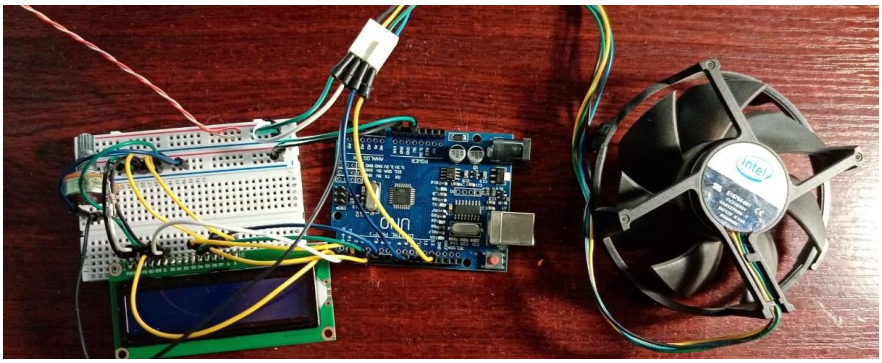

Рисунок 1 – Устройство в сборе

На рисунке видно, что вентилятор получает питание по отдельному контуру на макетной плате (витая пара белый-оранжевый подходит от источника тока 12 вольт). Однако можно реализовать одновременное питание. Для этого на Arduino платах предусмотрен бортовой стабилизатор напряжения. К нему подходит контакт, который обозначается на платах VIN (минус источника питания подключается к GND). Он позволяет питать проект на Arduino от 7-12 вольт.

На плату загружается следующая прошивка (рис. 2). Её разработка, происходит в Arduino IDE [6].

```
#include <LiquidCrystal.h>
#define pon_pin 5 // Аналоговый пин для потенциометра
#define cntrl_pin 3 // пин, управляющий скоростью вращения вентилятора

unsigned long lastflash;
int RPM;
LiquidCrystal lcd(0, 1, 7, 8, 12, 13);

void sens() {
  RPM = 60 / ((float)(micros() - lastflash) / 1000000); //расчет
  lastflash = micros(); //запомнить время последнего оборота
  }
void setup() {
  pinMode(2, INPUT);
  attachInterrupt(0, sens, RISING); //подключить прерывание на 2 пин при повышении сигнала
  lcd.begin(16, 2);
  lcd.setCursor(0,0);
  lcd.write("   FAN TESTER   ");
  delay(1000);
  lcd.clear();
  }
void loop() {
  if ((micros() - lastflash) > 1000000) { //если сигнала нет больше секунды
    RPM = 0; }  //считаем что RPM 0
  char buffer[32];
  lcd.setCursor(0,0);
       sprintf(buffer, "RPM: %d ob/min  ", RPM);
       lcd.print(buffer);
  int rawReading = analogRead(pon_pin);
  int volts = rawReading / 170.6; // получаем 6 положений для управлления вентилятором

  lcd.setCursor(0, 1);
    if(volts == 0){
      analogWrite(cntrl_pin, 0);
      lcd.print("FAN SPEED 10%"); }

    else if(volts == 1){
      analogWrite(cntrl_pin, 51);
      lcd.print("FAN SPEED 20%"); }

    else if(volts == 2){
      analogWrite(cntrl_pin, 102);
      lcd.print("FFAN SPEED 40%"); }

    else if(volts == 3){
      analogWrite(cntrl_pin, 153);
      lcd.print("FAN SPEED 60%"); }

    else if(volts == 4){
      analogWrite(cntrl_pin, 204);
      lcd.print("FAN SPEED 80%"); }

    else if(volts == 5){
      analogWrite(cntrl_pin, 255);
      lcd.print("FAN SPEED 100%"); }
  }
```

Рисунок 2 – Код прошивки

В данной прошивке используется стандартная для Arduino библиотека «LiquidCrystal». Она позволяет управлять различными дисплеями. Через константы заданы контакты для потенциометра (pon_pin) и для контакта управления скоростью вентилятора через ШИМ (cntrl_pin). В строках 5-7 создаются переменные для счёта количества оборотов вентилятора и инициализируется дисплей.

Для счёта количества оборотов в минуту, совершаемых вентилятором, создана функция sens. Она вызывается через аппаратные прерывания (строка 15). Функция setup инициализирует второй контакт как вход для датчика оборотов вентилятора. Он представляет собой датчик Холла, срабатывающий на каждый оборот вентилятора. Так же в setup на одну секунду надпись «FAN TESTER».

В функции loop, строкой 26 обнуляется значение для счётчика оборотов, если от него не было сигнала больше секунды. Вывод строки на дисплей, содержащий количество оборотов, осуществляется через функцию sprintf. В строках 29-30 обрабатываются значения, полученные с потенциометра. Функция «analogRead» возвращает значения из промежутка от 0 до 1023, он уменьшается до диапазона от 0 до 5. С 33 – 55 строку, через конструкцию «else if» прописываются значения для ШИМ на управляющем контакте вентилятора. Диапазон подаваемого значения находится на промежутке от 0 до 255. Для коэффициента заполнения 0, 20, 40, 60, 80, 100, посылаемое значение на контакт соответственно составляет 0, 51, 102, 153, 204, 255.

Работающее устройство показано на фото (рис. 3).

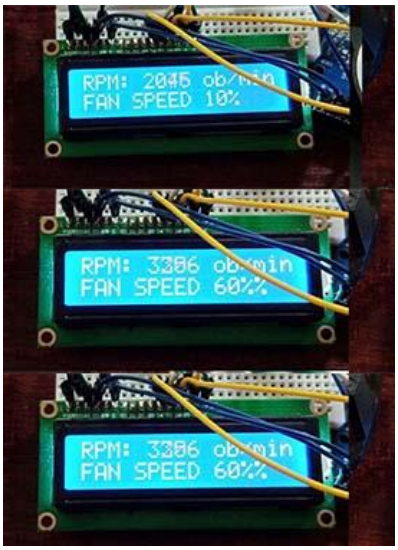

Рисунок 3 – Работающее устройство.

На рисунке видно, что для 0 количество оборотов вентилятора в минуту составляет примерно 2000, для 60% - 3200, для 100% - 3800. Удалось достичь диапазона в 1800 оборотов в минуту.

3. Выводы

Таким образом, было создано устройство на базе Arduino UNO, позволяющее тестировать различные вентиляторы, работающие от 12 вольт с управлением через ШИМ. Устройство считает количество оборотов в минуту. Управление происходит через потенциометр.

Библиографический список

1. Мишенин А.А., Лихолетов И.А. ШИМ-сигнал и ШИМ-контроллер, их назначение // Международная научно-техническая конференция молодых ученых БГТУ им. В.Г. Шухова, 2021. С. 3373-3375. URL: https://www.elibrary.ru/item.asp?id=46382617 (дата обращения: 1.11.2021).
2. Беляев В.П. Электропривод переменного тока с ШИМ-управлением // Энергетика. Известия высших учебных заведений и энергетических объединений СНГ, 2014. № 1. С. 11-20. URL: https://www.elibrary.ru/item.asp?id=22111674 (дата обращения: 1.11.2021).
3. Богданов К.В., Ловчиков А.Н. Моделирование преобразователей напряжения с ШИМ на языке Erlang // Актуальные проблемы авиации и космонавтики, 2012. № 1 (8). С. 348-349. URL: https://www.elibrary.ru/item.asp?id=22601680 (дата обращения: 1.11.2021).
4. Бабенко В.П., Битюков В.К. Имитационное моделирование ШИМ-регулятора // Фундаментальные проблемы радиоэлектронного приборостроения, 2015. № 15 (4). С. 150-153. URL: https://www.elibrary.ru/item.asp?id=25626151 (дата обращения: 1.11.2021).
5. Газизулин А.Ф. Интеллектуальный светодиодный светильник на платформе Arduino UNO // Прикладная электродинамика, фотоника и живые системы, 2019. С. 301-304. URL: https://www.elibrary.ru/item.asp?id=41467823 (дата обращения: 1.11.2021).
6. Arduino IDE. URL: https://www.arduino.cc/en/software (дата обращения: 28.10.2021).